



MULTIDISCIPLINARY AND
MULTIOBJECTIVE OPTIMIZATION
IN CONCEPTUAL DESIGN FOR
MIXED-STREAM TURBOFAN ENGINES

THESIS

Luc J.J.P. Nadon, Captain, CAF

AFIT/GAE/ENY/96D-6

19970205 031
035

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

DTIC QUALITY INSPECTED 3

DISTRIBUTION STATEMENT A

Approved for public release;
Distribution Unlimited

AFIT/GAE/ENY/96D-6

MULTIDISCIPLINARY AND
MULTIOBJECTIVE OPTIMIZATION
IN CONCEPTUAL DESIGN FOR
MIXED-STREAM TURBOFAN ENGINES

THESIS

Luc J.J.P. Nadon, Captain, CAF

AFIT/GAE/ENY/96D-6

Approved for public release; distribution unlimited

DTIC QUALITY INSPECTED 3

Disclaimer Statement

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense, the U.S. Government, the Canadian Department of National Defense, or the Canadian Government.

AFIT/GAE/ENY/96D-6

MULTIDISCIPLINARY AND
MULTIOBJECTIVE OPTIMIZATION
IN CONCEPTUAL DESIGN FOR MIXED-
STREAM TURBOFAN ENGINES

THESIS

Presented to the Faculty of the School of Engineering of the

Air Force Institute of Technology

Air University

In partial Fulfillment of the Requirements for the Degree of

Master of Science in Aeronautical Engineering

Luc Nadon, B.S.M.E.

Captain, CAF

December 1996

Approved for public release; distribution unlimited

Acknowledgment

This thesis would not have been possible without the outstanding support from a number of people whom I would like to take this opportunity to thank. First, I need to express my most sincere gratitude to LCol S. Kramer for directing me into research on optimization. Moreover, I wish to let him know how I appreciated his common sense approach to problems and his heartfelt moral support in dire straits. I am indebted to the committee members, Dr. P. I. King and Maj. E. Pohl, for the priceless advice they provided in their respective field. I would like to thank Mr. J. M. Stricker at Wright Laboratories for his help with the problem definition and his enthusiastic support for the project. I would also like to express my gratitude to Mr. R. E. Fredette, from Universal Technology Corporation, for the comprehensive Global Strike Aircraft data he provided. I need to recognize the invaluable support given by Dr. J. D. Mattingly, From Seattle University, for providing the ONX, OFFX and MISS computer original Fortran computer codes, without which the Matlab engine analysis codes could not have been created. Last, but not least, I want to thank K. Larsen, the engineering computer lab system manager, for her precious guidance in dealing with the many computer-related hurdles I encountered throughout this project.

Luc J.J.P. Nadon

Table of Contents

	Page
Acknowledgments	ii
List of Figures	vi
List of Tables	vii
List of Symbols	ix
Abstract	xiv
I. Introduction	1
Background	1
Problem Statement	3
Scope	3
Thesis Overview	4
II. Analysis Fundamentals	6
Introduction	6
General Optimization Concepts	6
Multidisciplinary (MDO) and Multiobjective (MOO) Optimization	9
Sequential Quadratic Programming	13
Matlab 'constr' SQP Optimization Code	16
Theory of Genetic Algorithms	16
General Concept	16
Constraints and Penalty Functions	18
GAOT GA Matlab Computer Code	20
On-Design Cycle Analysis	20
Off-Design Cycle Analysis	26
Mission Analysis	31
Takeoff Gross Weight (W_{TO}) Determination	35
Aircraft Cost Model	39

Engine Annulus Area Model	41
Miscellaneous constraint equations	44
Compressor Exit Pressure	44
Compressor Exit Temperature	45
Low Pressure Spool Rpm	45
High Pressure Spool Rpm	45
High Pressure Turbine Specific Work	45
Global Strike Aircraft and Short Range Interceptor	46
Global Strike Aircraft (GSA)	47
Short Range Interceptor (SRI)	49
 III. Methodology	 54
Introduction	54
Objectives	54
Assumptions	55
Optimization Problems Statements	56
Cases Investigated	60
Overall Taken to Achieve Thesis Objectives	62
Matlab as a Programming Environment	63
Creation of the On-Design Analysis Computer Code	64
Creation of the Off-Design Analysis Computer Code	66
Creation of the Mission Analysis Computer Code	69
On-Design Optimization Process	72
On-Design Objective Function	77
Optimization with Mission Process	79
W _{TO} Convergence	83
Mission Optimization Objective Function	88
 IV. Findings and Analysis	 92
Introduction	92
General Comments and Observations	92
Case 1: Test Case with Short Range Interceptor	98
Case 2: Short Range Interceptor On-design Mono-Objective Optimization	101
Case 3: Short Range Interceptor Mono-Objective Mission Optimization	103
Modified SRI Full Mission Optimization	109
Case 4: Global Strike Aircraft Mono-Objective Mission Optimization	110
Case 5: Global Strike Aircraft Multiobjective Mission Optimization	114
Case 6: Global Strike Aircraft Multiobjective On-Design Optimization	118

V. Conclusions and Recommendations	121
Introduction	121
Conclusions	121
Recommendations	125
Appendix A: Instructions on How to Obtain Optimization and Engine Analysis Computers Codes	126
Appendix B: Drag Profiles	127
Appendix C: Engine Optimization Computer Codes Operating Instructions	129
Bibliography	154
Vita	156

List of Figures

Figure	Page
1. Optimum Design Process	7
2. Two-Discipline MDO Process	10
3. Engine Reference Stations - Mixed Flow Turbofan	23
4. Off-Design Solution Scheme - Part One	29
5. Off-Design Solution Scheme - Part Two	30
6. Mission Analysis Flow Chart for a Leg	35
7. On-Design Optimization Process (gaondes.m)	76
8. On-design Global Optimization Objective Function Process (gafunbis.m)	78
9. On-Design Local Optimization Objective Function Process (fun.m)	79
10. Engine Optimization with Mission Process (gaoptmiss.m) - Part One	82
11. Engine Optimization with Mission Process (gaoptmiss.m) - Part Two	83
12. Global Mission Optimization Objective Function Process (gafunmiss.m)	89
13. Local Mission Optimization Objective Function Process (funmiss2.m)	90
14. C_{D0} vs. Flight Mach Number, M_0 , for the Short Range Interceptor	127
15. K_1 vs. Flight Mach Number M_0 , for the Short Range Interceptor	127
16. C_{D0} vs. Flight Mach Number, M_0 for the Global Strike Aircraft	128
17. K_1 vs. Flight Mach Number, M_0 , for the Global Strike Aircraft	128

List of Tables

Table	Page
1. On-Design Analysis Inputs, Outputs and Parameters.	22
2. Off-Design Analysis Inputs, Outputs and Parameters	31
3. Mission Analysis Legs	34
4. Global Strike Aircraft Mission Profile.	48
5. Global Strike Aircraft Optimization Data	49
6. Short Range Interceptor Mission Profile.	51
7. Short Range Interceptor Optimization Data	52
8. Trial-and-Error Process Short Range Interceptor Optimum Design.	53
9. Variables and Linear Scaling Boundaries for Case 1: SRI Test Case	99
10. Final Results for Case 1: SRI Test Case	100
11. Variables and Linear Scaling Boundaries for Case 2: SRI On-Design Optimization	102
12. Final Results for Case 2: SRI On-Design Optimization	103
13. Variables and Linear Scaling Boundaries for Case 3: SRI Mission Optimization	105
14. Final Results for the Case 3: SRI Mission Optimization	106
15. SRI Design Variables Values at $M = 1.6$ and $h = 35,000$ ft.	108
16. Variables and Linear Scaling Boundaries for Case 4: GSA Mono-Objective Mission Optimization.	111
17. Final Results for Case 4: GSA Mono-Objective Mission Optimization	112
18. Variables and Linear Scaling Boundaries for Case 5: GSA Multiobjective Mission Optimization	115

19. Final Results for Case 5: GSA Multiobjective Mission Optimization	115
20. Variables and Linear Scaling Boundaries for Case 6: GSA On-Design Optimization	119
21. Final Results for Case 6: GSA On-design Optimization.	120

List of Symbols

A	- Area (ft ²)
A*	- Area corresponding to M = 1 (ft ²)
a	- Speed of sound, coefficient in Γ function equation (ft/s)
b	- coefficient in Γ function equation
C ₁	- Engine cost coefficient (\$/lb thrust)
C ₂	- Avionics cost coefficient (\$)
C ₃	- Airframe cost coefficient (\$/lb empty weight)
C _D	- Drag coefficient
C _L	- Lift coefficient
C _p	- Specific heat at constant pressure (BTU/lbm-R)
C _v	- Specific heat at constant volume (BTU/lbm-R)
C _{TO}	- Power takeoff shaft coefficient
D	- Diameter (ft)
d	- Search direction
e _c	- Polytropic efficiency of fan
e _{cH}	- Polytropic efficiency of high pressure compressor
e _t	- Polytropic efficiency of turbine
e _{tH}	- Polytropic efficiency of high pressure turbine
e _{tL}	- Polytropic efficiency of low pressure turbine
F	- Uninstalled thrust, global objective function
f	- Fuel-to-air ratio of burner, objective function
f _{AB}	- Fuel-to-air ratio of afterburner
f ₀	- Overall engine fuel-to-air ratio
G	- Constraint penalty function
g	- Acceleration (ft/s ²), inequality constraint
H	- Hessian
h	- Altitude (ft), equality constraint, enthalpy
h _{PR}	- Heating value of fuel (BTU/lbm)
K	- Objective weight factor
K1	- Coefficient in lift-drag polar equation
K2	- Coefficient in lift-drag polar equation
k _{TO}	- Velocity ratio at takeoff
L	- Lagrangian function
M	- Mach number, Material modifier
MFP	- Mass flow parameter
m	- Mass flow rate (lbm/s)
m _{ci}	- Corrected mass flow at station I (lbm/s)
N	- Rotational speed (rpm), number of 360° turns
N _{eng}	- Number of engines
n	- Load factor
P	- Pressure (psi), power (W)

P_s	- Weight specific excess power (W/lb)
P_{TO}	- Power of takeoff shaft (W)
P_t	- Total pressure (psi)
q	- Dynamic pressure (lb/ft ²)
R	- Gas constant (BTU/lbm-R), parasitic drag (lb)
r	- Radius (ft), penalty parameter
rmc	- remaining fuel coefficient
S	- Uninstalled thrust specific fuel consumption (1/hr), wing area (ft ²)
sf	- Scaled objective function
T	- Temperature (R), installed thrust (lb)
t	- Time (s)
$TSFC$	- Installed thrust specific fuel consumption (1/hr)
T_t	- Total temperature (R)
U	- Discipline code
u	- Axial velocity (ft/s)
W	- Weight (lb)
W_E	- Empty weight (lb)
W_{eng}	- Engine weight (lb)
W_F	- Fuel weight (lb)
W_i	- Weight at beginning of mission leg (lb)
W_P	- Payload weight (lb)
W_{PE}	- Expended payload weight (lb)
W_{PP}	- Permanent payload weight (lb)
W_S	- Aircraft structural weight (lb)
W_{TO}	- Takeoff weight (lb)
x	- design variable
z_e	- Energy height (ft)
α	- Bypass ratio, thrust lapse, step length parameter
β	- Fuel fraction, bleed air fraction
Δ	- Change
δ	- Static pressure ratio (P/P_{SL})
ε_1	- Cooling air #1 mass flow rate fraction
ε_2	- Cooling air #2 mass flow rate fraction
ϕ	- Loss coefficient, global objective function with penalties
Γ	- Empty aircraft weight fraction
γ	- Ratio of specific heat
η	- Efficiency
η_{AB}	- Efficiency of afterburner
η_b	- Efficiency of burner
η_c	- Efficiency of fan
η_{cH}	- Efficiency of high pressure compressor
η_d	- Diffuser adiabatic efficiency

η_m	- Power transfer efficiency of shaft
η_{mH}	- Power transfer efficiency of high pressure spool
η_{mL}	- Power transfer efficiency of low pressure spool
η_{mP}	- Power transfer efficiency of power takeoff shaft
η_R	- Inlet total pressure recovery
η_{tH}	- Efficiency of high pressure turbine
η_{tL}	- Efficiency of low pressure turbine
λ	- Lagrange multiplier
Π	- Total pressure ratio, weight fraction
Π_{AB}	- Total pressure ratio of afterburner
Π_b	- Total pressure ratio of burner
Π_c	- Total pressure ratio of compressor
Π_{cH}	- Total pressure ratio of high pressure compressor
$\Pi_{c'}$	- Total pressure ratio of fan
Π_d	- Total pressure ratio of diffuser (inlet)
Π_{dmax}	- Total pressure loss in diffuser (inlet) due to friction
Π_M	- Total pressure ratio of mixer
Π_{Mmax}	- Total pressure loss in mixer due to friction
Π_n	- Total pressure ratio of nozzle
Π_r	- Isentropic freestream recovery pressure ratio
Π_{tH}	- Total pressure ratio of high pressure turbine
Π_{tL}	- Total pressure ratio of low pressure turbine
θ	- Static temperature ratio (T/T_{SL})
ρ	- Density (lbm/ft^3)
σ	- Static density ratio (ρ/ρ_{SL})
τ	- Total temperature ratio
τ_b	- Total temperature ratio of burner
τ_c	- Total temperature ratio of compressor
τ_{cH}	- Total temperature ratio of high pressure compressor
$\tau_{c'}$	- Total temperature ratio of fan
τ_d	- Total temperature ratio of diffuser (inlet)
τ_M	- Total temperature ratio of mixer
τ_{m1}	- Total temperature ratio of station 4 to 4a
τ_{m2}	- Total temperature ratio of station 4c to 4b
τ_n	- Total temperature ratio of nozzle
τ_r	- Adiabatic freestream recovery temperature ratio
τ_{tH}	- Total temperature ratio of high pressure turbine
τ_{tL}	- Total temperature ratio of low pressure turbine
τ_λ	- Enthalpy ratio of burner
ψ	- Normalized high pressure turbine inlet temperature (T_{t4}/T_{SL})

Subscripts

AB	- Afterburner
a	- Inlet annulus
av	- Available
b	- Burner
C	- Core flow
c	- Compressor, corrected
c'	- Fan
cH	- High pressure compressor
c1	- Cooling air #1
c2	- Cooling air #2
D	- Drag
dry	- No afterburning
e	- Exit
F	- Bypass flow
f	- Fuel, final
hub	- At engine hub
i	- Inlet, initial
M	- Mixer
m1	- Coolant mixer 1
m2	- Coolant mixer 2
max	- Maximum, with afterburner
min	- Minimum
mH	- Mechanical, high pressure spool
mL	- Mechanical, low pressure spool
mP	- Mechanical, power takeoff shaft
N	- iteration variable
n	- Nozzle
R	- Rotation, Reference
req	- Required
spec	- Specific
STALL	- Corresponding to stall
TO	- Takeoff, power takeoff
TR	- Transition
t	- Turbine, total
tip	- At tip of blades
TH	- Overall efficiency
tH	- High pressure turbine

tL - Low pressure turbine
wet - With afterburning
0→10 - Engine station location

λ - isentropic freestream recovery

Superscripts

()* - Corresponding to $M = 1$

Abstract

Despite major advances in design tools such as engine cycle analysis software and computer-aided design, conceptual gas turbine engine design is essentially a trial-and-error process based on the experience of engineers. Modern optimization concepts, such as multidisciplinary optimization (MDO), and multiobjective optimization (MOO), linked with sequential quadratic programming (SQP) methods and genetic algorithms (GA), were applied to the conceptual engine design process to automate the conceptual design phase. Robust integrated computer codes were created to find the optimal values of eight engine parameters in order to minimize fuel usage, aircraft cost and engine annulus area over a given mission. The engine cycle selected for study was the mixed-stream, low-bypass turbofan. SQP and GA optimization algorithms were integrated with on-design and off-design engine cycle analysis and mission analysis computer codes created by the authors to obtain the optimized conceptual engine design for an imaginary short range interceptor and the Global Strike Aircraft U.S. Air Force concept. The process used a non-specific approach that can be applied to a wide variety of missions and aircraft. All the codes were written in Matlab, and so operate under the same programming architecture and can be easily upgraded or modified.

MULTIDISCIPLINARY AND MULTIOBJECTIVE OPTIMIZATION IN CONCEPTUAL DESIGN FOR MIXED-STREAM TURBOFAN ENGINES

I. Introduction

Background

The aircraft engine design and manufacturing fields are extremely competitive since multimillion, if not multibillion, dollar contracts are involved. Small variations in the selection of engine design parameters may have a significant impact on the thrust available, the quantity of fuel consumed over a mission, or the costs of the engine and aircraft. Unfortunately, engine design is an inexact science that involves a trial-and-error process based on engineers' and corporate experience. This is due to the fact that a gas turbine engine is a sophisticated device which involves complex interactions of factors such as 3-D viscous and turbulent flows, material stresses, and thermodynamics, to name a few. The design process searches for a design that meets all thrust requirements at the lowest cost and fuel consumption. This process is based on empirical models (obtained through experience and experiments) and hopefully leads to an overall lighter and cheaper aircraft (15: 12.1-12.3).

This search in a complex environment for the best engine design is well suited for multidisciplinary optimization (MDO) and multiobjective (MOO) approaches. Optimization involves the determination of optimal values for a given set of design

variables (such as turbine inlet temperature and compressor pressure ratio) to minimize or maximize a given function (such as fuel consumption). It does so by performing searches for minima (or maxima) over the design space with the use of numerical algorithms.

MDO involves the application of multiple engineering fields, such as aerodynamics and materials, in the optimization of complex design problems. In the aerospace world, MDO has been successfully applied to the structures and controls field. MDO is a relatively new tool in aircraft engine design and includes such disciplines as thermodynamics, aerodynamics, material stresses and cost analysis. Multiobjective optimization is concerned with the integration in the objective function of multiple objectives to minimize (maximize). These objectives may span one or many disciplines. Most efforts in this promising field have concentrated on the optimization of only a few engine design parameters or the optimization was carried out over only a few flight conditions, or for a specific aircraft. What is needed is an integrated tool that can be used for any number of variables and over a wide range of engine types and flight conditions.

As the function to be minimized (or maximized) becomes more complex, and as the number of design variable increases, the location of the optimal design point becomes increasingly hard to determine and many algorithms simply cannot converge toward an optimal solution. Among the most powerful calculus-based optimization approach is sequential quadratic programming (SQP). SQP converge to a local minima (or maxima) with the use of quasi-Newton and quadratic programming methods. An example of SQP algorithm is the Matlab 'constr.m' code (6). Another promising (and somewhat recent) technique used in these situations is the genetic algorithm (GA). Genetic algorithms apply

aspects of biological genetic theory, such as reproduction and mutations, to select optimal design through an evolutionary process. One such algorithm is GAOT, developed at North Carolina State University (7). Both GA and SQP were integrated into a single algorithm in the present project to exploit strengths of each technique. These optimization tools are used in conjunction with mission analysis, on-design and off-design cycle analysis codes created by the author, based on methods by Mattingly (10), in order to obtain an optimal conceptual engine. The model thus created is modular and meant to be improved and updated by considering additional disciplines such as stresses and installation losses.

Problem Statement

The need exists to improve the trial-and-error conceptual engine design process. A conceptual engine design should meet all mission and flight requirements with the lowest fuel usage, cost, and engine size. The methodology selected to solve the aforementioned problem must consider a modular optimization model to allow for the inclusion of additional factors such as stresses, installation losses, and advanced cost models. As examples, optimal conceptual designs will be developed for the DOD Global Strike Aircraft concept (4) and for an imaginary short range interceptor previously investigated by the author using a trial and error process.

Scope

This project applied a multidisciplinary and multiobjective optimization approach and used sequential quadratic programming and genetic algorithms in conjunction with

engine cycle and mission analysis codes, and simple aircraft cost and engine size models to develop an engine design process applicable to optimized engine uninstalled performance over a whole mission, from takeoff to landing. The engine cycle and mission analysis codes will be called as part of the objective function. The engine cycle under study was the mixed-stream, low bypass turbofan. It was selected because of its increasing use for military aircraft applications.

Thesis Overview

The engine design optimization process is developed in the following chapters of this thesis:

Chapter II. The theory and analytical tools used in the optimization procedure are covered. Concepts like optimization, genetic algorithms, sequential quadratic programming methods, on-design and off-design cycle analysis and mission analysis are briefly discussed. This chapter also includes descriptions of the computer codes used and the characteristics of the Global Strike Aircraft and the short range interceptor.

Chapter III. This chapter summarizes the methodology in solving the optimization problem. Thesis objectives and assumptions are stated and the optimization problem is formulated in details. The optimization processes for both the single flight condition and the full mission cases are described.

Chapter IV. The project findings and analysis are covered in this chapter. It describes the results obtained for both the Global Strike Aircraft and the short range interceptor. Problems and discrepancies are discussed at this point.

Chapter V. An overview of the optimization process, including its results, is provided.

Conclusions and recommendations are presented. The thrust of future research based on this project is discussed.

II. Analysis Fundamentals

Introduction

This chapter covers the theory and tools used in this thesis. These include optimization, sequential quadratic programming and genetic algorithms, engine cycle analysis, and mission analysis. All the engine theory introduced is applied to a mixed-stream, low-bypass turbofan cycle. Also introduced therein are the Global Strike Aircraft and the short range interceptor.

General Optimization Concepts

The purpose of optimization is to find the values of design variables to minimize (or maximize) a given function. Simply put, the process search for the function minima (or maxima). The overall process is described in Figure 1. The terms ‘objective function’, ‘design variables’, and ‘constraints’ need to be defined at this point.

The objective function, also called cost function, is the function to be minimized (or maximized). Objective functions may be equations or, in the case of complex problem, outputs from computer codes. The variables that describe the problem and define a given design are the design variables. Design constraints are the limitations imposed on the design variables or the objective function to ensure a feasible design is within given resources and requirements (1:22).

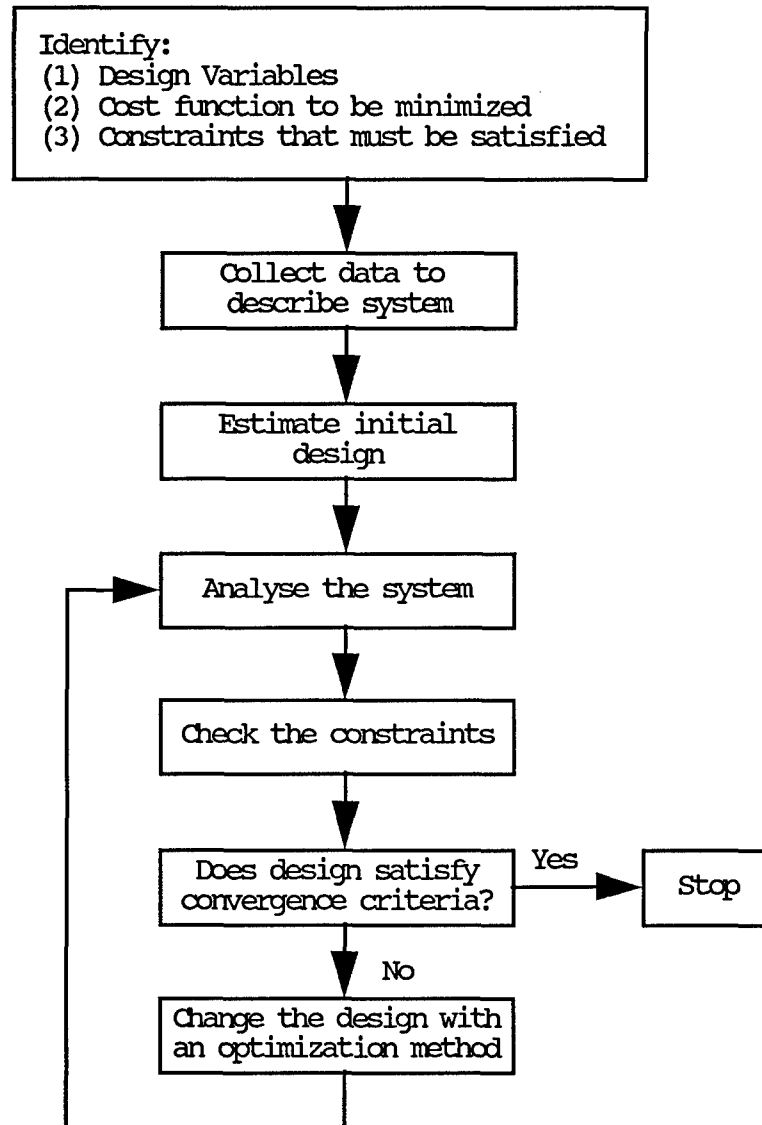


Figure 1. Optimum Design Process (1:9)

Simple unimodal functions that involve a small number of variables use calculus based algorithm to find the optimum point. However, in the case of complex, non-smooth functions with multiple minima and large number of design variables, the search for the global optimum design requires more powerful computer algorithms. One of the most

effective calculus-based constrained optimization algorithm is sequential quadratic programming (SQP). SQP is the method used in this thesis and is described in more details below. Another extremely powerful tool is the genetic algorithm which is also discussed below.

The standard optimization problem formulation is as follows (1:45):

$$\begin{aligned}
 &\text{minimize } f(\mathbf{x}) = (x_1, x_2, x_3, \dots, x_n) & (1) \\
 &\text{Subject to: } h_j(\mathbf{x}) \equiv h_j(x_1, x_2, x_3, \dots, x_n) = 0 \\
 &\quad \quad \quad j = 1 \text{ to } p \\
 &\quad \quad \quad g_i(\mathbf{x}) \equiv g_i(x_1, x_2, x_3, \dots, x_n) \leq 0 \\
 &\quad \quad \quad i = 1 \text{ to } m
 \end{aligned}$$

where $f(\mathbf{x})$ is the objective function, ' \mathbf{x} ' is a design variables vector of size n , $h_j(\mathbf{x})$ represent p equality constraints and $g_i(\mathbf{x})$ represents m inequality constraints. All constraints are standardized to the form ' $= 0$ ' or ' ≤ 0 ' to simplify optimization algorithms. A constraint of the form ' $a \leq g_k(\mathbf{x}) \leq b$ ' is called a side constraint and it defines the boundaries for a given design variables. Side constraints are expressed in standard form by breaking them down into two constraints, ' $-g_k(\mathbf{x}) + a \leq 0$ ' and ' $g_k(\mathbf{x}) - b \leq 0$ '. A constraint is considered active if the value of the variable or parameters of interest has hit the constraint limit, i.e. $g(\mathbf{x}) = 0$. Active constraints are a good indicator of the factors that limits the performance of a given design as they indicate which constraints actively restrict improvements on a design.

Maximization problems are transformed into minimization problems by the multiplication of the objective function by -1; i.e. the maximization of $z(\mathbf{x})$ is treated as the minimization of $f(\mathbf{x}) = -z(\mathbf{x})$.

Multidisciplinary (MDO) and Multiobjective Optimization (MOO)

The purpose of MDO is to find the optimum values for a set of design variables dependent on functions from various engineering disciplines. This implies that the output of one or more disciplines are required as input to one or more other disciplines and vice-versa. The various disciplines may or may not share the entire set of design variables.

Figure 2 summarizes the MDO process for a two-discipline problem. One can see that the output for each discipline's code, U_1 and U_2 , feed into the other code and into the optimizer (3). The optimizer, in turn, evaluates new values for the design variable vector, X_D , that are used as input for both discipline codes. This process is repeated until an optimum design is reached that satisfies the requirements of both disciplines.

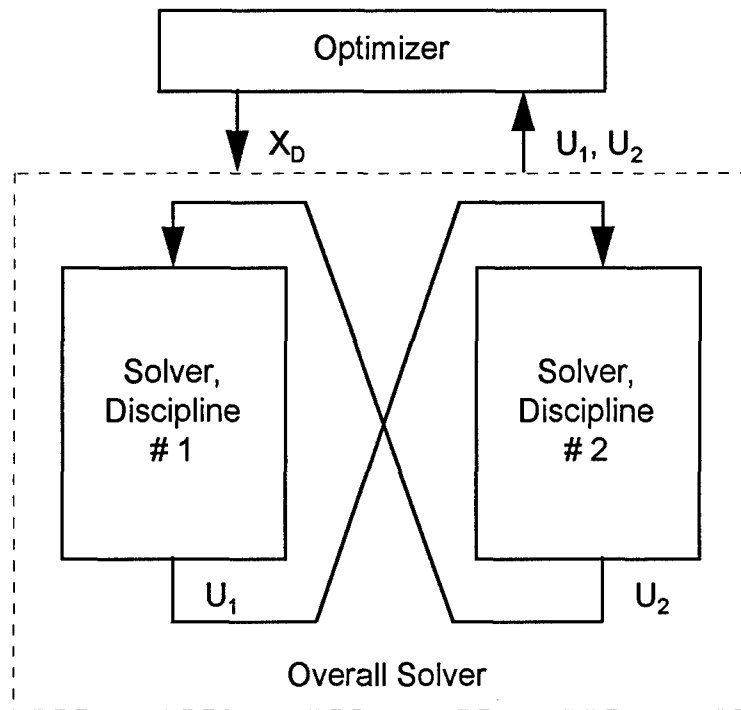


Figure 2. Two-Discipline MDO process (3:767)

The goal of multiobjective optimization is to determine an optimum design that involves the minimization (or maximization) of multiple objective functions. More often than not, these various functions have conflicting objectives, such as the maximization of performance and the minimization of costs. MOO may or may not involve functions from different disciplines. A simple approach to MOO is to create a global objective function as follows:

$$F(x) = K_1sf_1(x) + K_2sf_2(x) + \dots + K_nsf_n(x) \quad (2)$$

where

$F(x)$ = global objective function

K_k = objective weight factors

f_k = individual objective functions

sf_k = scaled objective function values

n = number of objectives

The value of each individual objective function is scaled in order to ensure that the value of each term of the global function is of the same order of magnitude. This is necessary to avoid terms with high magnitude having a disproportionate impact on the global function. For example, if the value of one objective is expressed in thousands of thrust pounds of thrust and another objective is expressed in millions of dollars, the cost term would dominate, even though cost might not be the most critical term of the global function.

Linear scaling, with individual objective values scaled between zero and one, was selected for this thesis mainly because of its simplicity. Linear scaling consists in the use of linear expressions to determine the scaled value of each individual. The linear expressions are of the form $y=ax+b$ and are based on expected minimum and maximum values (based on experience) of a given objective, and on how desirable these values are. The desirable limit is assigned a value of one and the undesirable value is assigned a value of zero. For example, if a given objective is total aircraft cost and the cost of the aircraft is expected to vary between \$10,000,000 and \$30,000,000, the \$10,000,000 limit is

assigned a value of one while the \$30,000,000 is assigned a value of zero since a low cost is more desirable than a high cost. With the objective limits and their desirability determined, the scaled value for the given objective is expressed as follows:

$$sf_k = \frac{1}{(\text{desir} - \text{undesir})}(f_k - \text{undesir}) \quad (3)$$

where

desir = value of the given objective desirable limit

undesir = value of the given objective undesirable limit

Equation (3) is a linear expression. The process described above is repeated for each term of the global objective function. It is important to note that the minimization of a variable becomes a maximization problem when this variable is linearly scaled since high scaled values are desirable.

The weight factors K_k are assigned by the designer(s) to vary the impact of each objective on the global objective function. A relatively high value of K means that the objective has a higher impact or priority and vice-versa. K_k values are assigned by experience and must take into account the ultimate goal of a design (i.e. for example, is it more important to have high performance or a low cost?). Integer K_k values were used in this thesis. For example, if the first objective of a global function is assigned $K_1=2$ and the second objective is assigned $K_2=1$, then the first objective has twice the impact of the

second one. The default is $K_k=1$ for all three objectives (fuel usage, cost , and area), which mean that each objective has an equal impact on the global objective function.

Sequential Quadratic Programming

SQP is an advanced nonlinear programming method. The technique attempts to mimic Newton approaches by using quasi-Newton methods to approximate the Hessian of the Lagrangian at each iteration of the optimization process. These approximations are used to generate a quadratic subproblem (QP) which, in turn, is used to determine the search direction for a line search (6). The Lagrangian function is expressed as follows:

$$L(\bar{x}, \lambda) = f(\bar{x}) + \sum_{i=1}^m \lambda_i g_i(\bar{x}) \quad (4)$$

where

L = Lagrangian

f = objective function

λ_i = Lagrange multipliers. Represent sensitivity to given constraints

g_i = constraints

x = design variable

m = number of constraints

The QP subproblem is based on a quadratic approximation of the Lagrange function and by the linearization of nonlinear constraints. It can be solved using any QP algorithms. The QP sub problem may be expressed as:

$$\begin{aligned}
 \text{minimize} \quad & 1/2 \mathbf{d}^T \mathbf{H}_k \mathbf{d} + \nabla f(\mathbf{x}_k)^T \mathbf{d} \\
 & \mathbf{x} \in \mathcal{R} \\
 & \nabla \mathbf{g}_i(\mathbf{x})^T \mathbf{d} + \mathbf{g}_i(\mathbf{x}) = 0 \quad i = 1, \dots, m_e \\
 & \nabla \mathbf{g}_i(\mathbf{x})^T \mathbf{d} + \mathbf{g}_i(\mathbf{x}) \leq 0 \quad i = m_e + 1, \dots, m
 \end{aligned} \tag{5}$$

where

\mathbf{d} = search direction vector

\mathbf{H} = Hessian matrix of the Lagrangian

k = iteration counter

m_e = number of equality constraints

A popular method to approximate the Hessian is the Broyden-Fletcher-Goldfarb-Shanno (BFGS) method. It updates the Hessian as follows:

$$\mathbf{H}_{k+1} = \mathbf{H}_k + \frac{\mathbf{q}_k \mathbf{q}_k^T}{\mathbf{q}_k^T \mathbf{s}_k} - \frac{\mathbf{H}_k^T \mathbf{H}_k}{\mathbf{s}_k^T \mathbf{H}_k \mathbf{s}_k} \tag{6}$$

where

$$\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$$

$$\mathbf{q}_k = \nabla f(\mathbf{x}_{k+1}) + \sum_{i=1}^m \lambda_i \nabla g_i(\mathbf{x})_{k+1} - \left(\nabla f(\mathbf{x}_k) + \sum_{i=1}^m \lambda_i \nabla g_i(\mathbf{x})_k \right)$$

The updated Hessian and the QP problem solution are used to obtain a new iterate of the design point:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k \quad (7)$$

The step length parameters, α_k , is chosen to minimize $f(\mathbf{x})$ along the given direction \mathbf{d} , subject to the constraints, and used to determine the amount of decrease of the objective function for an iteration. It is determined by a line search in such a way as to obtain a sufficient decrease in a merit function as described in Chapter 2 of Reference 6. Chapter 2 of this reference also covers SQP and BFGS methods in more details. One drawback of SQP methods is their requirements for a feasible initial point to start. This situation can be resolved with the solution of a linear programming problem that involves the minimization of constraints' slack variables (6:2-24). Once a feasible point has been located with the linear programming method, the algorithm may proceed with the SQP phase.

Matlab 'constr' SQP Optimization Code

The Matlab subroutine 'constr.m' is a sequential programming algorithm. It solves a quadratic subproblem at each iteration. At each iteration, an estimate of the Hessian is obtained with the BFGS technique. The line search is performed with the merit function described in Chapter 2 of Reference 6. The 'constr.m' code is discussed in details in Reference 6.

Theory of Genetic Algorithms (GA)

Genetic algorithms are relatively new in the world of optimization. As their name implies, GAs apply the biological principles of genetics and evolution to select an optimum design through a survival-of-the-fittest process. Included in this process are reproductions, crossovers and mutation of design points. These algorithms are not the most efficient as they require numerous iterations and function evaluations. On the other hand, they are quite robust, they are good at finding global minima and they perform well with multiple-minima, non-smooth functions. A major difference with other optimization method are the stochastic elements, discussed below, that are integrated into the process (5: 1-6).

General Concept. A basic GA starts with the selection of a scheme to code the values of the variables for each design in a manner suitable for genetic manipulations. The second step involves the creation of a population of properly coded random points that

cover the design space, and the determination of objective function values for each design. Binary codes are a good example of such a coding scheme.

With the objective function values, the probability that a given design is selected to the next generation is determined with the expression $P_s = f_i / \sum f$, where P_s is the selection probability, f_i is the design objective function value and $\sum f$ is the sum of the objective function values for all the current design. A generation consists of one iteration of the genetic algorithm process as performed on the current population. A random experiment is performed a number of times equal to the population size to select which designs, and how many of each, will survive to the next generation.

The members of this new population are mated (i.e. paired) randomly. A given percentage of these pairs will be selected at random for crossover. For each pair of designs thus created (and selected), a crossover point is selected, again at random. The crossover point indicates the point at which 'gene swapping' will occur. In the case of a binary code, the crossover point indicates a bit position. For a given mated pair, the bits to the right of the crossover point design (in the case of a binary code) are exchanged between the pair's members. This creates two 'offspring', i.e. two new designs which are a combination of the 'parents' characteristics.

The set of offspring is the new population of designs to be used for the next iteration. The process is repeated over numerous generations until the global optimum design is identified. The GA process ensures that unsuitable designs (those with low selection probabilities) are eliminated. A low mutation rate, of the order of 0.001 per bit per generation or lower, is integrated into the algorithm to ensure that promising parts of

the design space are not neglected in the selection process. A detailed discussion of the GA process, illustrated with an example, is included in Reference 5, Chapter 1. Although many variations exists, all genetic algorithms follow the basic method described above.

Constraints and Penalty Functions. One drawback of standard GAs is their inability to deal with constraints other than side constraints. A common approach to remedy to this situation is to use penalty functions. Penalty functions are terms added to the basic objective function and whose values depend on the margin by which constraints are violated. With the use of penalty functions, the initial constrained problem is converted into an unconstrained problem by considering a function of the form (14:487):

$$\phi_k = f(x) + r_k \sum_{j=1}^m G_j[g_j(x)] \quad (8)$$

where

ϕ = global function that includes basic objective function and penalties

k = iteration number

$f(x)$ = basic objective function

r_k = penalty parameter

G_j = penalty function for a given constraint

$g_j(x)$ = constraints

m = number of constraints

G_j is a penalty applied to the basic objective function and depends on the value for a given constraint. Its magnitude is dependent r_k and on the amount by which a constraint is violated. The further away a design is from the feasible space, the greater is the penalty. An unviolated constraint has a penalty of zero. The total penalty applied to the basic function is the sum of all the individual penalties. There are many forms of penalty functions. For use in this thesis, the author developed the following penalty function:

$$G_j = \left(2 * \left(\frac{\text{const} + \text{limit}}{\text{limit}} \right) \right)^2 \quad (9)$$

where

const = constraint value for a given design

limit = constraint limit for a given constraint

The variable 'const' is a positive value that represents the margin by which a design failed to meet a constraint, while 'limit' is the constraint boundary. For example, let's have a design constraint that require the engine area to be smaller than 10 ft². This constraint is then expressed as 'const = (Area - 10 ft²) ≤ 0'. The value '10 ft²' represent the value for the 'limit' variable. Any design that meets the constraint will have a negative value for 'const'. If 'const' is positive, then the area is greater than 10 ft², and the design must be rejected. So, if an unacceptable design has an area of 12 ft², the value of 'const' would be 'const = (12 - 10) = 2'.

Function (9) was selected because it performed well for the problem at hand and also because it scaled all constraints to an order of magnitude readily adaptable to a linearly scaled objective function. Equation (9) is independent of the iteration number and values for r_k were set to one.

GAOT GA Matlab Computer Code

The GA computer optimization code used for this project is called GAOT. It has been developed at the North Carolina State University (7) and has been created as a Matlab subroutine. It uses binary coding for integer representation and floating point representation for problems that involve real numbers. Contrary to other basic GAs, GAOT is not limited to positive objective function values. However, this algorithm is design for function maximization and thus any minimization problem requires a sign change. GAOT is covered in details in Reference 7.

On-Design Cycle Analysis

The purpose of on-design engine cycle analysis is to determine the engine parameters and the flight conditions (Mach number and altitude) for which the engine is optimized, i.e. the point at which the engine is the most efficient. In this project the cycle of interest is the mixed stream, low bypass turbofan cycle, the analysis of which is presented in detail in Chapter 4 of Reference 10. It must be understood that since an engine is designed for one flight condition, for example Mach 0.8 at 30,000 ft, the engine will operate off design for a significant part of a given mission (landing, climb, low level

flight, etc.). It is thus of critical importance to select an on-design point that insures acceptable performance over the entire aircraft flight envelope.

For a given set of engine parameters and flight conditions, such as overall compressor pressure ratio (Π_c), fan pressure ratio (Π_{c^*}), bypass ratio (α), altitude, mass flow (m_0) and Mach number, the cycle analysis will determine the values for, among others, uninstalled thrust (F), specific fuel consumption (SFC), and specific thrust (F/m_0). The process goes through the following steps:

- a. Preliminary computations based on flight conditions, parameters, such as values for R , Π_d , and τ_r .
- b. Fan and high pressure compressor properties, such as pressure and temperature ratios, and efficiencies.
- c. Burner, coolant mixers and turbine properties.
- d. Engine exhaust mixer properties.
- e. Evaluation of overall engine performance.

For the cycle of interest, on-design analysis consist of solving the system of equations listed in Appendix E of Reference 10. The inputs and outputs involved for the on-design cycle analysis are included in Table 1, while Figure 3 illustrates the engine stations reference numbering.

Table 1. On-Design Analysis Inputs, Outputs and Parameters (10:467)

<u>Inputs</u>	
Flight parameters:	$M_0, T_0(R), P_0(\text{psia})$
Aircraft system parameters	β, C_{TO}
Design limitations:	
Perfect gas constants:	$\gamma_c, \gamma_b, \gamma_{AB}, C_{pc}, C_{pt}, C_{pAB} \text{ (BTU/lbm-R)}$
Fuel heating value:	$h_{PR} \text{ (BTU/lbm)}$
Component figures of merit:	$\epsilon_1, \epsilon_2,$ $\Pi_b, \Pi_{dmax}, \Pi_{Mmax}, \Pi_{AB}, \Pi_n,$ $e_c', e_{cH}, e_{tH}, e_{tL},$ $\eta_b, \eta_{AB}, \eta_{mL}, \eta_{mP}, \eta_{mH}$
Design choices:	$\Pi_c, \Pi_c', \alpha, T_{t4}, T_{t7} (R), M_5, P_0/P_9$
<u>Outputs</u>	
Overall performance:	$F/m_0 \text{ (lbf/lbm/s)}, S \text{ (1/hr)}, f_0,$ $\eta_P, \eta_{tH}, V_9/V_0, P_{t9}/P_9, T_9/T_0$
Component behavior:	$\Pi_{tH}, \Pi_{tL}, \Pi_M,$ $\tau_c', \tau_{cH}, \tau_{tL}, \tau_{tH}, \tau_\lambda, \tau_{\lambda AB},$ $f_{AB}, f,$ $\eta_c', \eta_{cH}, \eta_{tH}, \eta_{tL},$ M_5, M_6, M_9

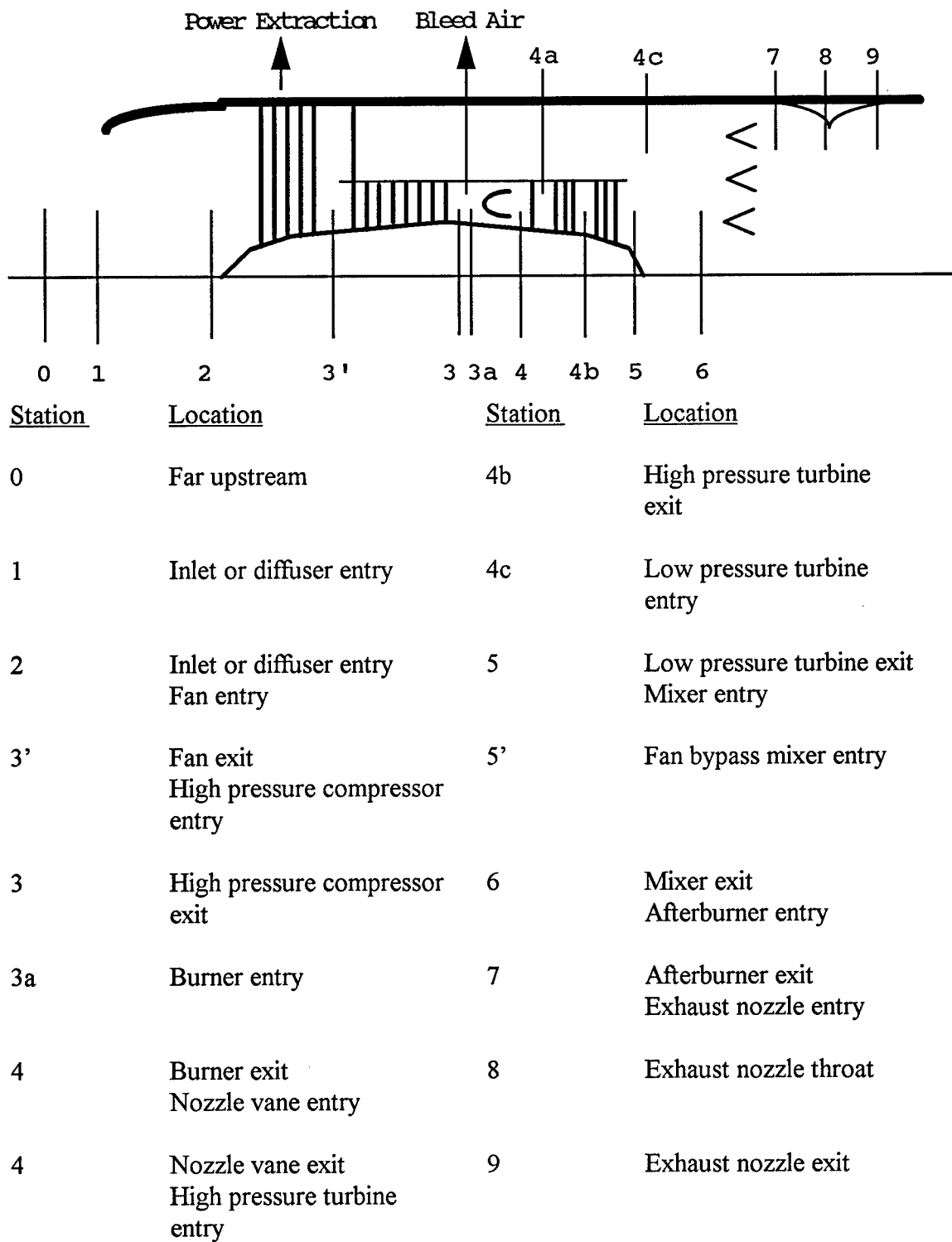


Figure 3. Engine Reference Stations - Mixed Flow Turbofan (10:99)

The cycle model described in Chapter 4 and Appendix E of Reference 10 is based on the assumptions quoted as follows:

- The flow is on average steady.
- The flow is one-dimensional at the entry and exit of each component and at each axial station.
- The fluid behaves as a calorically perfect gas with constant C_p , C_v , and γ across the diffuser, fan, compressor, turbine, nozzle, and connecting ducts.
- Values of C_p , C_v , γ , and R can change across the burner, mixer, and afterburner.
- The total pressure ratio of the diffuser or inlet is

$$\Pi_d = \Pi_{dmax} \cdot \eta_{Rspec}$$

where Π_{dmax} is the total pressure ratio due only to wall friction effects and η_{Rspec} is the ram recovery of military specification MIL-E-5008B as given by:

$$\eta_{Rspec} = 1 \quad \text{for} \quad M_0 \leq 1$$

$$\eta_{Rspec} = 1 - 0.075(M_0 - 1)^{1.35} \quad \text{for} \quad M_0 \geq 1$$

- The fan is driven by the low pressure turbine, which also provides the mechanical power for accessories, P_{TO} .

- The high pressure compressor receives air directly from the fan, and is driven by the high pressure turbine.
- High pressure bleed air and cooling air are removed between station 3 and 3a.
- Flow in the bypass duct (from station 3' to 5') is isentropic.
- The effect of cooling on turbine efficiency must be accounted for by a reduction of e_{tH} due to m_{c1} and m_{c2} .
- The fan and core streams mix completely in a mixer, the actual total pressure ratio Π_M being the value for an ideal constant area mixer Π_{Mideal} multiplied by Π_{Mmax} or

$$\Pi_M = \Pi_{Mideal} \cdot \Pi_{Mmax}$$

where

Π_{Mideal} is the total pressure ratio across an ideal constant area mixer

Π_{Mmax} is the total pressure ratio due only to wall friction effects

(10:106-107)

The Matlab m-files 'onx.m' and 'onxopt.m', were created by the author to perform the on-design cycle analysis described above to be used, respectively, as a stand-alone program, and as a function routine for part of the optimization process. Instructions on how to obtain the codes are included in Appendix A. The on-design cycle analysis is critical for this thesis as it is a part of the objective function used to search for optimal engine designs.

Off-Design Cycle Analysis

Once the on-design analysis has been completed, it is necessary to evaluate the performance of the baseline engine at operating conditions other than design. It is the role of the off-design cycle analysis to determine the performance of the engine over the flight envelope. Although there is practically an infinite number of potential operating conditions for a given engine, only the significant off-design points where the engine is expected to operate are investigated. As a starting point, a given aircraft mission profile, as obtained from an aircraft Request for Proposal or other defining documents, provides the designer with a series of off-design operating conditions in the form of mission legs. Mission legs include such flight conditions as turns, takeoff, cruise, acceleration, etc. Examples of mission profiles are included below, in the section that describes the Global Strike Aircraft and the short range interceptor.

The model used in this thesis for off-design analysis is described in details in Chapter 5 of Reference 10. The solution of the off-design problem involves the evaluation of 14 dependent variables with 14 independent equations through the use of an iterative process illustrated in Figures 4 and 5. The aforementioned equations are described in Reference 10, Appendix F. The inputs and outputs involved for the off-design cycle analysis are included in Table 2.

The off-design process is presented in Figure 4 and 5. The outputs are the same as for the on-design analysis and they apply to uninstalled performance. A major difference with on-design analysis is the requirement for an iterative approach to converge to an off-design solution. The model used for this thesis used the low pressure turbine temperature

ratio (τ_{tL}), the mixer bypass ratio (α') and the mass flow (m_0) as the controlling iteration variables. Iterations on M_5 , P_{t5}/P_{t5} and $M_{5'}$ are included in the algorithm to ensure that these values stay within an acceptable range. With a process similar to the one described for the on-design analysis, turbine and compressor properties are evaluated to converge to the appropriate τ_{tL} value. Exhaust mixer properties are evaluated in order to iteratively determine α' and m_0 . The whole iterative scheme is repeated until a solution has been found or until it is determined that no feasible solution exists, i.e. values for τ_{tL} , α' or m_0 do not converge. The final step is to determine overall engine performance at the off-design point. This model is based on assumptions quoted as follows:

- The flow is choked at the high pressure turbine entrance nozzles (station 4), at the low pressure turbine entrance nozzles (station 4c), and at the exhaust nozzle (station 8). The case of the unchoked exhaust nozzle is also included in this analysis.
- The component efficiencies and total pressure ratios, η_c , η_{cH} , η_b , η_{tH} , η_{tL} , η_{AB} , η_{mL} , η_{mH} , η_{mP} , and Π_b , Π_{Mmax} , Π_{AB} , Π_n do not change from their design values.
- Bleed air and cooling air fractions are constant. Power takeoff is constant. Leakage effects are neglected.
- Gases are assumed to be calorically perfect both upstream and downstream of the burner and afterburner and values of γ_t , C_{pt} , γ_{AB} , C_{pAB}

do not vary with throttle setting. However variations of γ_6 and C_{p6} due to the variation of the bypass ratio are included.

- The exit area (A_9) of the exhaust nozzle is adjustable so that the pressure ratio P_0/P_9 can be set to predetermined value.
- The area at each engine station is constant. However the area of station 8 changes with the afterburner setting.
- The diffuser total pressure ratio, Π_d , is given by Eqs. (4.8a), (4.8b) and (4.8c) (10:138).

Note also that since values of f and f_{AB} are small when compared to unity, these variables are ignored in expressions with terms of the order of one or larger.

Off-design analysis is required for this project to ensure that the optimum design would operate successfully over the given aircraft mission. The Matlab m-files 'offx.m' and 'offxmiss.m', created by the author, perform off-design calculations at all flight conditions with the model described above. The 'offx.m' file is used as a stand-alone code while the 'offxmiss.m' file is part of the mission analysis code. This latter program is used to help define the design space with the application of constraints that represent the off-design mission legs. Instructions on how to obtain the off-design codes are included in Appendix A. The codes also contains provisions to throttle back T_{t4} if the pressure and temperature at the exit of the high pressure compressor (P_{t3} and T_{t3}) are beyond set limits. Limits are also imposed on the rotational speeds of both the low-pressure and high pressure spools.

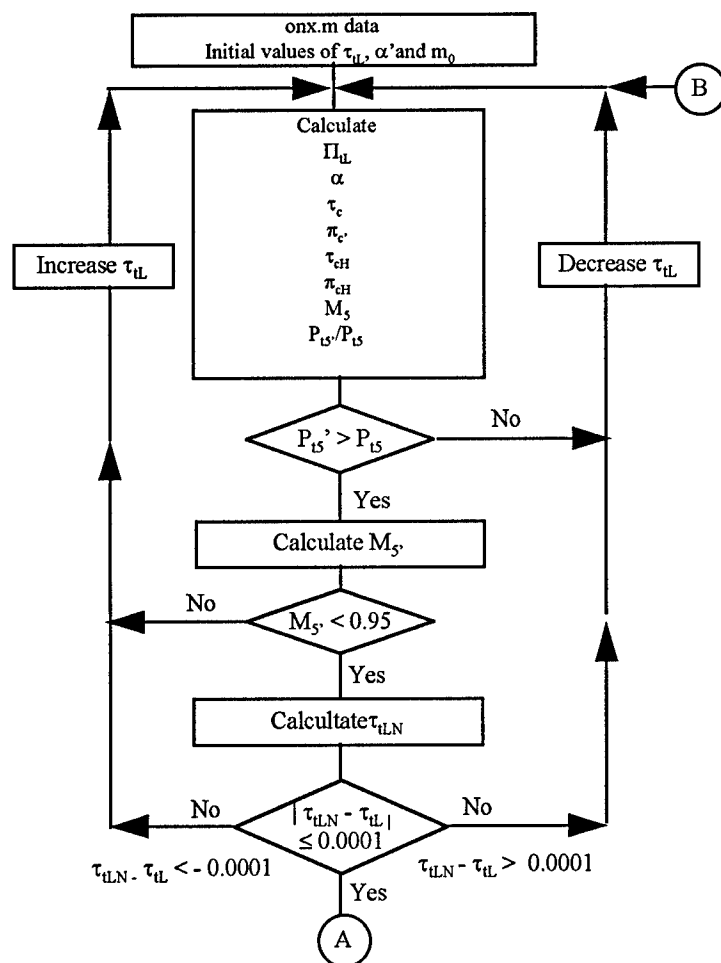


Figure 4. Off-Design Solution Scheme - Part One.
Adapted from Fig. 5.2 (10:151)

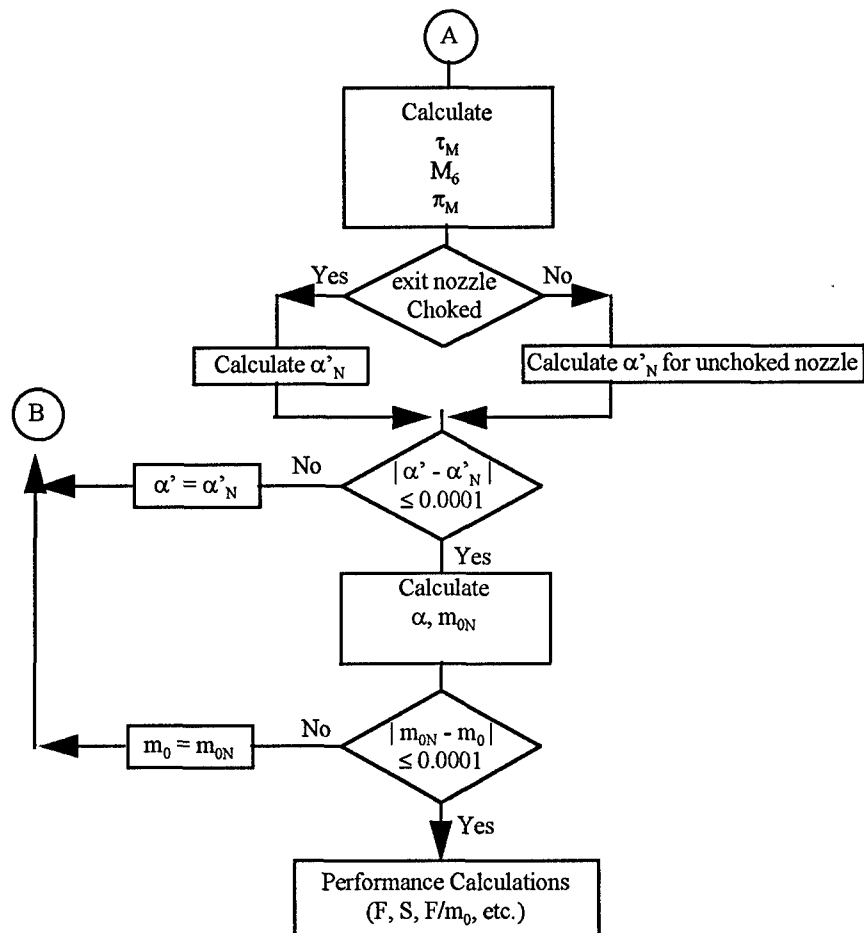


Figure 5. Off-Design Solution Scheme - Part Two.
Adapted from Fig. 5.3 (10:152)

Table 2. Off-Design Analysis Inputs, Outputs and Parameters (10:473)

<u>Inputs</u>	
Off-design choices	
Flight parameters:	$M_0, T_0(R), P_0(\text{psia})$
Throttle setting:	$T_{t4} (^{\circ}R), T_{t7} (^{\circ}R)$
Exhaust nozzle setting:	$P_0/P_9 (A_8/A_{8R} \text{ if unchoked})$
Design constants	
Π 's:	$\Pi_{tH}, \Pi_b, \Pi_{dmax}, \Pi_{Mmax}, \Pi_{AB}, \Pi_n$
τ 's:	$\tau_{m1}, \tau_{tH}, \tau_{m2}$
η 's:	$\eta_b, \eta_{AB}, \eta_{c'}, \eta_{mP}, \eta_{mH}, \eta_{cH}, \eta_{tL}$
Gas properties:	$\gamma_c, \gamma_t, \gamma_{AB}, C_{pc}, C_{pt}, C_{pAB} (\text{BTU/lbm-R})$
Others:	$\beta, \epsilon_1, \epsilon_2, A_5'/A_5, h_{PR} (\text{BTU/lbm})$
Reference conditions	
Flight parameters:	$M_{0R}, T_{0R}(R), P_{0R}(\text{psia}), \tau_{tR}, \Pi_{tR}$
Component behavior:	$\Pi_{dR}, \Pi_{c'R}, \Pi_{cHR}, \Pi_{tLR}, \Pi_{MR},$ $\tau_{c'R}, \tau_{cHR}, \tau_{tLR}, \tau_{MR}$
Others:	$\alpha_R, \alpha'R, \Gamma_{8R},$ $(A/A^*)_{5R}, (P_{t5}/P_{t5})_R, C_{TOR}$
<u>Outputs</u>	
Overall performance:	$F (\text{lb}_f), m_0 (\text{lbm/s}), S (1/\text{hr}), f_0,$ $\eta_P, \eta_{tH}, \alpha, V_9/V_0, P_{t9}/P_9, T_9/T_0$
Component behavior:	$\Pi_{c'}, \Pi_{cH}, \Pi_{tL}, \Pi_M,$ $\tau_{c'}, \tau_{cH}, \tau_{tL}, \tau_M, \tau_\lambda, \tau_{\lambda AB},$ $f, f_{AB},$ $M_5, M_{5'}, M_6, M_9$

Mission Analysis

The primary purpose of a mission analysis is to determine the total fuel usage, W_f , of an aircraft for a complete mission from takeoff to landing. The mission is expressed as a mission profile, which is basically a single flight broken down in phases (or legs), that a

given engine design must meet. Examples of typical legs are takeoff, climb, cruise, turn, and acceleration phases. It is a critical tool since it also verifies that a given engine design meets the mission, leg by leg. The main ways the mission analysis algorithm will consider a given engine has having failed a mission are insufficient thrust, unchoked turbines and bypass and core flow not mixing properly. Mission analysis is also used to estimate W_{TO} .

The mission analysis process selected for this project is described in details in Chapter 3 of Reference 10. Before a mission analysis may proceed, values for wing loading (W_{TO}/S), thrust loading (T_{SL}/W_{TO}) and an initial estimate for W_{TO} must be provided. In order to determine the fuel weight variation for each leg, flight conditions are categorized into two types, based on their weight specific excess power (P_s) status. Flight conditions where $P_s > 0$, such as acceleration are classified as type A, while conditions where $P_s = 0$, such as cruise, are classified as type B. The fuel weight determination process is quoted from Reference 9 as follows (9:13-14):

The aircraft weight is simply a combination of the empty weight, fuel weight and payload weight

$$W_{TO} = W_E + W_f + W_P \quad (A)$$

The rate of change of the aircraft weight as a function of fuel consumption is

$$dW/dt = - dW_f/dt = - TSFC \times T \quad (B)$$

where TSFC is the installed thrust specific fuel consumption and T is the installed thrust. This equation can be rewritten as

$$dW/W = -TSFC(T/W)dt \quad (C)$$

The desire is to determine appropriate expressions for various legs of the mission which relate the initial and final weight for that phase. These expression are of the form

$$W_f/W_i \leq 1 \quad (D)$$

and Eq (C) is used to develop the weight fraction equations which corresponds to flight regimes where $P_s > 0$ (type A), or $P_s = 0$ (type B).

Type A behavior is exhibited during constant speed climb, horizontal acceleration, climb and acceleration, and takeoff acceleration. Type B behavior relates to constant altitude/speed cruise and turn, best cruise Mach number and altitude, loiter, warm-up, takeoff rotation and constant energy height maneuvers. The developed equations are

Type A:

$$\frac{W_f}{W_i} = \exp \left\{ -\frac{C\sqrt{\theta}}{V(1-u)} \Delta \left(h + \frac{V^2}{2g_0} \right) \right\} \quad (E)$$

Type B:

$$\frac{W_f}{W_i} = \exp \left\{ -C\sqrt{\theta} \left(\frac{D+R}{W} \right) \Delta t \right\} \quad (F)$$

C in Equations (E) and (F) is the specific fuel consumption, θ is the static temperature ratio, u is the total drag-to-thrust ratio and D and R are drag terms. Note that the weight fraction W_f/W_i is also expressed as ${}_i\Pi_j$ for a given leg, where subscripts i

and j relate, respectively, to the subscripts i and f described above. The final weight fraction for a given mission is obtained by the multiplication of all Π_i together.

The various potential mission analysis legs are listed in Table 3. The weight fraction equations for each of these legs are discussed in Chapter 3 of Reference 10.

Figure 6 describes the mission analysis process for a given leg.

Table 3. Mission Analysis Legs

- | | |
|----------------------------------|-----------------------------------|
| - Constant speed climb | - Loiter |
| - Horizontal acceleration | - Warm-up |
| - Climb and acceleration | - Takeoff rotation |
| - Takeoff acceleration | - Constant energy height maneuver |
| - Constant altitude/speed cruise | - Deliver expendables |
| - Constant altitude/speed turn | - Descend ($P_s < 0$) |
| - Best cruise Mach and altitude | |

The Matlab m-files 'miss.m' and 'gamiss.m', developed by the author, performs mission analysis as described above for an engine design provided by the on-design code. Instructions on how to obtain the codes are included in Appendix A. The 'miss.m' file is used as a stand-alone code while the 'gamiss.m' file is part of the mission optimization process. In order to keep the model simple, the analysis includes constant installation losses of 9.1%, the default suggested by Mattingly (11:10). Since most legs occur at off-design flight conditions, the mission analysis code includes 'offxmiss.m' in order to evaluate engine performance for each leg. The 'mission.m' code, along with 'onx.m' and 'offx.m', is part of the objective function required to evaluate the engine design over an entire mission. The code includes provisions to throttle back T_{17} and T_{14} as required to

ensure the available thrust is equal to the required thrust. Moreover, the code can evaluate the final W_{TO} (with the method described below) and determine the feasibility of a maximum Mach, maximum altitude leg.

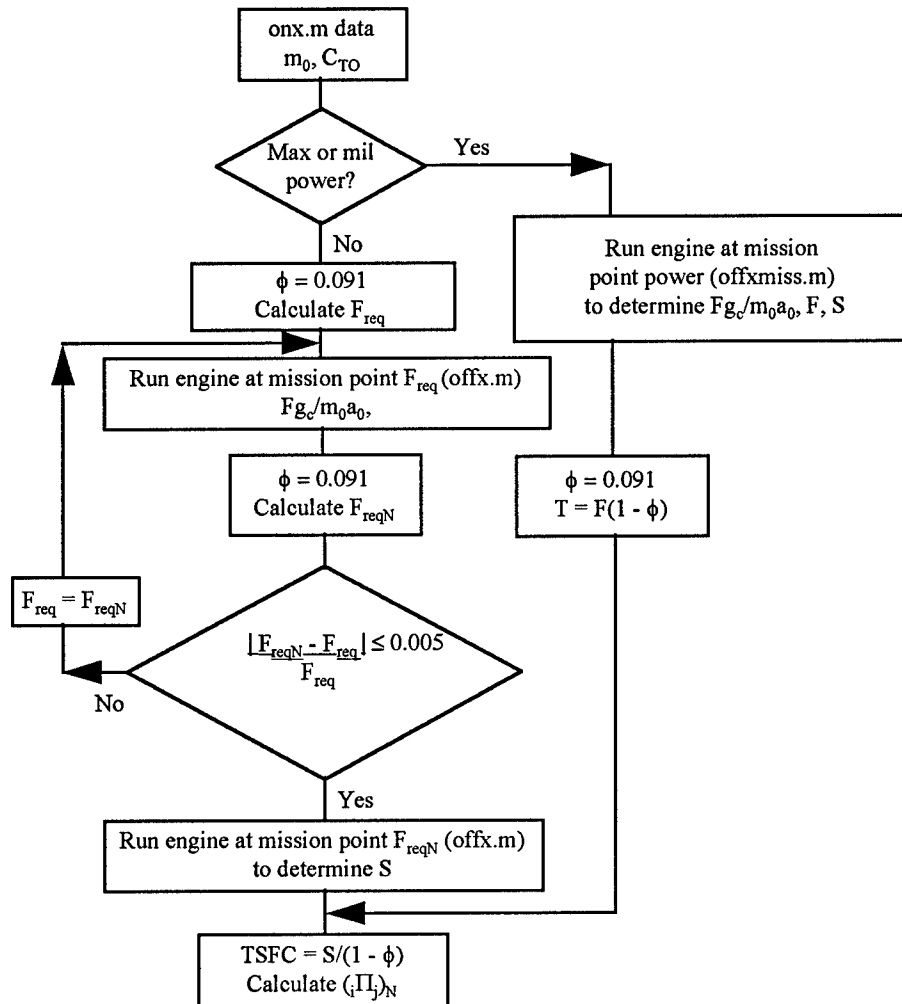


Figure 6. Mission Analysis Flow Chart for a Leg.
Adapted from Fig 6.E3 (10:209)

Takeoff Gross Weight (W_{TO}) Determination

As part of the optimization process, the value for W_{TO} has to be updated as the fuel weight W_f varies, since changes in W_f directly affect W_{TO} . The three methods

developed in this study to update W_{TO} are described below. It must be realized that the weight models and the mission analysis code are based on fixed values of wing loading, W_{TO}/S and thrust loading, T_{SL}/W_{TO} .

The first approach investigated is valid for all aircraft and is discussed in more details in Reference 8. In this method, as a first step, the W_{TO} value selected for the mission analysis is compared with

$$W_{TO(\Gamma)} = \frac{W_P + (rmc \times W_f)}{1 - \frac{W_f}{W_{TO}} - \Gamma} \quad (10)$$

where

$W_{TO(\Gamma)} = W_{TO}$ as a function of empty weight fraction (lb)

W_P = total payload weight (lb)

rmc = remaining fuel coefficient = % of W_f remaining on landing

W_{TO} = present value of W_{TO} as provided to the mission analysis code (lb)

Γ = empty aircraft weight fraction, W_E/W_{TO}

Γ is tailored to a given aircraft type and is further defined as (10:70):

$$\Gamma = MaW_{TO(miss)}^b \quad (11)$$

where

M = material modifier

a, b = coefficients determined by the type of aircraft

The coefficient a and b are determined by experience for a given aircraft type and have no specific physical meaning. Since one of the aircraft investigated in this study, the Short Range Interceptor, is a combat aircraft built with advanced materials, values for M , a and b are selected as 0.9, 2.34 and -0.13 respectively (10:70,80).

If $W_{TO(miss)} \approx W_{TO(\Gamma)}$, then the right value for W_{TO} has been obtained. If not, additional iterations of the mission analysis, with new estimates for $W_{TO(miss)}$, are performed until convergence. The author found that a good way to estimate $W_{TO(miss)}$ for successive iterations was to use this relationship:

$$\text{estimate } W_{TO(miss)} = (W_{TO(miss)} + W_{TO(\Gamma)})/2 \quad (12)$$

The method described above was found to be inadequate for larger aircraft, such as the Global Strike Aircraft. It was noted that any variation in W_f larger than a few hundred pounds from one GA generation to the next would make W_{TO} diverge to either zero or infinity due to the highly non-linear behavior of Equation (11).

In order to correct the limitations of the weight model described above, a second approach, based on data from Table 8.2 from Reference 4, was developed for the GSA using the following linear relationship between W_{TO} and W_f :

$$W_{TOGSA} = 1.3138634W_f + 39621 \quad (13)$$

It must be noted that Equation (13) is only valid for the GSA and that a new W_{TO} model would have to be developed for aircraft which do not fit the models described therein. It must be noted that the two weight models described above are not depended on a given aircraft empty weight. The final W_{TO} obtained from optimization run using these models can be used to help determine the lowest possible aircraft structural weight.

A third W_{TO} model was developed in order to investigate means to accelerate gross weight convergence and reduce the dependence of the W_{TO} convergence scheme on aircraft data. This additional scheme use fuel weight as a ninth design variable, i.e. $x_9 = W_f$. This new variable is a dummy variable which is not a part of the engine design as such. This $x_9 = W_f$ variable is used to determine an individual W_{TO} for every design and to ensure the gross weight does not diverge to the infeasible mission region. The side constraints boundaries for x_9 are defined by the user and are based on available data or experience. The gross weight for each design investigated by the global optimizer is determined as follows:

$$W_{TO} = W_{empty} + W_{fdes} \quad (14)$$

where

W_{empty} = Aircraft takeoff weight without fuel.

W_{fdes} = dummy fuel weight variable = x_9

W_{empty} includes the weights of the structure, avionics, payload and engines. A good estimate of this value can be obtained from available aircraft data. A constraint is also applied to ensure that only the gross weight does not diverge upward:

$$W_f \leq W_{fdes} \quad (15)$$

where

W_f = mission fuel weight as determined by the mission analysis program

If x_9 is properly bounded by side constraints, Equation (15) ensures that W_f will stay below the upper fuel weight boundary and thus keep W_{TO} from diverging upward. This method was investigated with the SRI. It must be realized that since W_{empty} is constant, this weight model is more applicable to engine retrofit situation since the aircraft structural weight is not adjusted as W_f varies.

Aircraft Cost Model

One of the objectives to minimize as part of the optimization process is the aircraft cost. The cost model is a term of the global objective function. The model used is a simple approximation based on empirical data (16) and defined as follows:

$$Cost_{A/C} = C_1 \cdot F_{SL} \cdot N_{eng} + C_2 + C_3 \cdot W_s \quad (16)$$

where

$Cost_{A/C}$ = total aircraft cost (\$)

C_1 = engine cost coefficient (\$/lb uninstalled thrust)

C_2 = avionics cost (\$)

C_3 = airframe Cost coefficient (%/lb aircraft empty weight)

F_{SL} = uninstalled thrust at sea level static per engine (lb)

W_s = aircraft structural weight (lb)

N_{eng} = number of engine

C_1 , C_2 and C_3 are constants derived from experience. Typical values for C_1 and C_3 are between 60 and 120 \$/lb, and 500 to 700 \$/lb respectively (16). C_2 values depend on the function of the aircraft and covers a wider cost range. The variable W_s needs to be broken down further:

$$W_s = W_{TO} - W_f - W_P - W_A - W_{eng} \quad (17)$$

where

W_{TO} = aircraft takeoff weight (lb)

W_f = fuel weight (lb)

W_P = total payload weight (lb)

W_A = avionics weight (lb)

W_{eng} = engine weight (lb)

It is necessary for the airframe designer(s) to provide values for W_{TO} initial, W_A and W_P . The last term of Equation (17), W_{eng} , is determined as follows (16):

$$W_{eng} = N_{eng} F_{SL} \frac{(W_{eng} / m_0)}{F_{SL} / m_0} \quad (18)$$

where

m_0 = engine mass flow (lbm/s)

The parameter (W_{eng}/m_0) represents the specific engine weight and is a indicator of an engine relative size. A typical value of 8 for this parameter will be used for this project (16). It must be realized that for the weight convergence model based on Equations (14) and (15), W_s is essentially a constant term in Equation (16), since the values of W_{empty} , W_A and W_P are fixed.

The cost model discussed above is crude and should not be considered accurate to determine aircraft cost. This is so because this model does not include such important aspects of overall system costs such as research and development, logistic support or the use of state of the art technology and materials. This model should eventually be replaced by a more accurate and complex cost model or computer code. However, an accurate estimate of the aircraft cost is not important at this point, as one only needs to determine the relative cost of each design. The main purpose of including cost is to explicitly allow for the possible direct tradeoff of aircraft cost and engine performance as measured by the total fuel usage over a mission. For this reason, the cost model used in this study is considered adequate for the problem at hand.

Engine Annulus Area Model

The third objective to achieve as part of the optimization process is to ensure the engine is as small as possible. In this project, it is achieved through the minimization of

the inlet annulus area A_a , as it is a good indicator of engine size (13:section 10-4). The annulus area corresponds to the area of the inlet where the airflow goes through, i.e. the annular area between the hub and the tip of the first stage fan rotor. From this point onward, this area will be referred as the inlet area. The area model presented below is described in details in Reference 13, section 10-4. A_a is defined as follows:

$$A_a = \pi(r_{tip}^2) \left[1 - \left(\frac{r_{hub}}{r_{tip}} \right)^2 \right] \quad (19)$$

where

A_a = annulus area (ft²)

r_{tip} = radius at tip of the first stage fan rotor (ft)

r_{hub}/r_{tip} = hub to tip ratio

In terms of engine performance parameters, the specific thrust (F_{SL}/m_0) is expressed as follows:

$$\frac{F_{SL}}{m_0} = \frac{F_{SL} \sqrt{\theta}}{m_{0spec} \delta A_a} \quad (20)$$

From Equation (20) an expression for A_a in terms of engine performance can be derived:

$$A_a = \frac{F_{SL} \sqrt{\theta}}{m_{0spec} \delta (F_{SL} \cdot m_0)} \quad (21)$$

where

θ = static temperature ratio

δ = static pressure ratio

$m_{0\text{spec}} = (m_0\sqrt{\theta})/(\delta A_a) = \text{specific flow (lbm/sec-ft}^2\text{)}$

From Equation (19) we can derive an expression for r_{tip} :

$$r_{\text{tip}} = \sqrt{\frac{A_a}{\pi \left[1 - \left(\frac{r_{\text{hub}}}{r_{\text{tip}}} \right)^2 \right]}} \quad (\text{ft}) \quad (22)$$

In this project, Equation (21) is a term of the global objective function while Equation (22) is used to define a constraint on maximum engine size with a limitation on maximum engine diameter (r_{tip}). Values for $m_{0\text{spec}}$ and hub-to-tip ratio are selected by the designer. It is important to realize that $m_{0\text{spec}}$ is dependent on Mach number. It peaks at Mach 1.0 and drops off at speeds away from that point. Typical values for $m_{0\text{spec}}$ and $r_{\text{hub}}/r_{\text{tip}}$ are between 36 and 40 lbm/sec-ft² and 0.35 respectively (16). These values are used in the optimization process.

It is interesting to note that the minimization A_a is equivalent to an optimization on m_0 since for all things being equal, the smallest engine would be the one meeting all requirements with the lowest mass flow. This coupling of m_0 and A_a was observed in all the cases investigated in Chapter 4. This indicates that the engine area term of the cost

function could be replaced by the value for m_0 with virtually no effect on the final design. A_a could then be evaluated as part of the final performance calculations for the optimal design.

Miscellaneous Constraint Equations

As part of the optimization process, constraints must be imposed for P_{t3} , T_{t3} , spools rpm and the specific work for the high pressure turbine stage ($\Delta h_t/\psi$). Constraints on P_{t3} , T_{t3} and rpm are imposed due to material stresses considerations. The turbine specific work constraint is discussed in more details below.

The expression for the aforementioned variables are as follows:

Compressor Exit Pressure (10:Ch 4):

$$P_{t3} = (P_0 \Pi_r \Pi_d \Pi_c \Pi_{ch}) / 144 \quad (23)$$

where

P_{t3} = compressor exit pressure (psi)

P_0 = freestream pressure (psi)

Π_r = isentropic freestream recovery pressure ratio

Π_d = diffuser pressure ratio

Π_c = fan pressure ratio

Π_{ch} = high pressure compressor pressure ratio

Compressor Exit Temperature (10:Ch 4):

$$T_{t3} = T_0 \tau_r \tau_{c'} \tau_{cH} \quad (24)$$

where

T_{t3} = compressor exit temperature (R)

T_0 = freestream temperature (R)

τ_r = isentropic freestream recovery temperature ratio

$\tau_{c'}$ = fan temperature ratio

τ_{cH} = high pressure compressor temperature ratio

Low Pressure Spool Rpm (12):

$$RPM1 = 100 \frac{\Pi_{cH}}{\Pi_{cHR}} \sqrt{\frac{T_0 \tau_r}{T_{0R} \tau_{rR}}} \quad (25)$$

where

RPM1 = rpm low pressure spool (% of on-design rpm)

R = subscript referring to on-design values of the appropriate variable

High Pressure Spool Rpm (12):

$$RPM2 = 100 \frac{\Pi_{cH}}{\Pi_{cHR}} \sqrt{\frac{T_0 \tau_r \tau_{c'}}{T_{0R} \tau_{rR} \tau_{c'R}}} \quad (26)$$

where

RPM2 = high pressure spool rpm (% on-design rpm)

High Pressure Turbine Specific Work (16):

$$\frac{\Delta h_t}{\psi} = C_{pt} T_{t4} \eta_{tH} \left(1 - \Pi_{tH}^{\frac{\gamma_t - 1}{\gamma_t}} \right) / \left(\frac{T_{t4}}{T_{SL}} \right) \quad (27)$$

where

$\Delta h_t/\psi$ = high pressure turbine specific work (BTU/lbm)

ψ = Normalized high pressure turbine inlet temperature (T_{t4}/T_{SL})

C_{pt} = specific heat at constant pressure downstream of burner (BTU/lbm-R)

T_{t4} = burner exit temperature (R)

η_{tH} = high pressure turbine efficiency

γ_t = ratio of specific heat downstream of burner

The constraint on $\Delta h_t/\psi$ is imposed because of aerodynamics considerations.

Since, in Equation (27), C_{pt} and T_{SL} are constants and since η_{tH} does not vary much over the allowable Π_{tH} range, it turns out that Π_{tH} is the critical variable in the determination of the high pressure turbine specific work. This implies that Equation (27) is essentially a constraint on the high pressure turbine pressure ratio. Historical data suggest that for state-of-the-art engine, the practical lower limit for Π_{tH} is of the order of 0.35. Below this value, major shock losses and flow separation occurs, thus limiting the maximum work

that can be extracted by a single high pressure turbine stage (16). This lower limit of Π_{HT} translate into a value for $\Delta h_t/\psi$ of the order of 32 BTU/lbm. This was the value used for this study. The practical upper limit for Π_{HT} is approximately 0.52, a value beyond which the high pressure turbine might be unchoked (8).

Global Strike Aircraft and Short Range Interceptor

In this section we introduce the sample aircraft used to evaluate the performance of the SQP and GAs in finding optimum engine designs. The aircraft selected are the Global Strike Aircraft and a short range interceptor. The descriptions of both aircraft include general overviews of their roles, typical mission profiles, and the parameters values and constraints used in this thesis.

Global Strike Aircraft (GSA). The GSA is a new aircraft concept under study at Wright Laboratories. It is described in detail in Reference 4. It is a very long range stealthy attack aircraft designed to takeoff from friendly bases, preferably in the continental United States, far away from the battle zone and then fly at high altitude to take out specific targets with highly accurate standoff guided munitions. A typical mission would involve flight over 5000 nm in supersonic cruise at altitudes on the order of 60,000 ft. It is an interesting aircraft to study because the level of engine performance required for this aircraft is based on forecast of propulsion technology available by the year 2025. The mission profile is presented in Table 4, while the data necessary for the optimization process is included in Table 5. The GSA drag profile is included in Appendix B.

Table 4. Global Strike Aircraft Mission Profile (4)

<u>Leg</u>	<u>Altitude (ft)</u>	<u>Mach Number</u>	<u>Comments</u>
1. Warm up, Taxi	0	0	Idle
2. Takeoff	0	0.4	Takeoff distance: 3,600ft, Mil power
3. Accelerate to climb speed	0	0.4-0.8	Mil power
4. Climb and accelerate to supercruise	0-60,000	0.8-1.5	Mil power, minimum fuel path
5. Outbound supercruise	60,000	1.5	Part mil power, distance 5,000nm
6. Combat			
6a. Weapon launch sweep	60,000	1.5	Part mil power
6b. Launch Weapons			
6c. 720° turn	60,000	1.5	Mil power, turns at 1.5g
7. Return supercruise	60,000	1.5	Part mil power distance 5,000nm
8. End loiter and land	0	Best Endurance Mach	Part mil power, Loiter at 30,000 ft landing distance 3,600 ft

Most of the general data and constraints information included in Table 5 were obtained in Reference 4 and 16. Most of the engine parameters are the default values proposed by Mattingly (10:116, 125). A few of the selected values, however, require further elaboration. The K_i weight factors are set to one, thus giving each of the objectives (fuel consumption, cost and weight) an equal impact on the design. Values of C_1 and C_3 were set at 60 \$/lb F_{SL} and 500\$/lb W_E respectively to represent the state of the art (16). The values for h_f , e_c , e_m , bleed air fraction, η_{AB} and the P_{15} constraints were set

in Reference 16. The Π_{UL} constraint was set to keep both the high and low pressure turbines choked (8).

Table 5. Global Strike Aircraft Optimization Data (4,10,16)

<u>General Data</u>	<u>Engine Parameters</u>	<u>Constraints</u>
$K_1 = 1$	$C_{pc} = 0.238 \text{ BTU/lbm-R}$	$T_{i4} \leq 4,460^\circ \text{ R}$
$K_2 = 1$	$C_{pt} = 0.295 \text{ BTU/lbm-R}$	$T_{i7} \leq 0^\circ \text{ R (no AB)}$
$K_3 = 1$	$\gamma_c = 1.4$	$m_0 \leq 500 \text{ lbm/s}$
$C_1 = 60 \text{ \$/lb thrust}$	$\gamma_t = 1.3$	$0.2 \leq M_5 \leq 0.7$
$C_2 = 40,000,000$	$h_f = 18,400 \text{ BTU/lbm}$	$\Delta H_T/\theta \leq 32 \text{ BTU/lbm}$
$C_3 = 500 \text{ \$/lb } W_E$	Cooling air #1 = 0.05	$0.2 \leq M_5 \leq 0.95$
$W_{eng}/m_0 = 8$	Cooling air #2 = 0.05	Spools RPM $\leq 110\%$
$m_{0spec} = 40 \text{ lbm/sec-ft}^2$	$\Pi_B = 0.97$	Cost $\leq 150,000,000 \text{ \$}$
$r_{hub}/r_{tip} = 0.35$	$\Pi_{Dmax} = 0.97$	$r_{tip} \leq 4 \text{ ft}$
$W_{TO \text{ initial}} = 259,500 \text{ lb}$	$\Pi_N = 0.98$	$1 \leq \alpha \leq 3$
$W_{f \text{ initial}} = 167,345 \text{ lb}$	$e_c = 0.91$	$3 \leq \Pi_c \leq 8.5$
$W_P = 16,440 \text{ lb}$	$e_{ch} = 0.9$	$80 \leq \Pi_c \leq 100$
$W_A = 3,180 \text{ lb}$	$e_{tH} = 0.89$	$\Pi_{UL} \leq 0.5$
$W_{PE} = 16,440 \text{ lb}$	$e_{tL} = 0.91$	$P_{i5} \geq P_{i5}$
No of engines = 4	$e_{mH} = 0.99$	Takeoff distance $\leq 3,600 \text{ ft}$
$r_{mc} = 0.05$	$e_{mL} = 0.99$	$P_{i3} \leq 2,000 \text{ psi}$
$T_{SL}/W_{TO} = 0.391$	$e_{mPTO} = 0.99$	$T_{i3} \leq 2,260^\circ \text{ R}$
$W_{TO}/S = 51.77 \text{ lb/ft}^2$	$\eta_{AB} = 0.95$	$F_{SL} \geq 25,343 \text{ lb}$
$K1 \text{ and } C_{D0} = \text{Appendix B}$	$\gamma_{AB} = 1.3$	$W_f \leq 175,000 \text{ lb}$
$K2 = 0$	$C_{PAB} = 0.295$	
$W_{empty} = 92,155 \text{ lb}$	$\Pi_{mix \text{ max}} = 0.97$	
	$P_0/P_9 = 1$	
	Bleed air = 0.005	
	$\Pi_{AB} = 0.96$	
	$\eta_b = 0.98$	

Short Range Interceptor (SRI). The short range interceptor is a hypothetical concept investigated by the author as part of a term project for conceptual engine design class. It is a small, high thrust and highly maneuverable fighter aircraft with supercruise

capability optimized for air combat. The conceptual engine design for this aircraft has been performed in detail using the trial-and-error process outlined in Chapters 1 to 7 of Reference 10. As the SRI engine cycle parameters had been obtained through the trial-and-error optimization process described at Reference 10, the short range interceptor engine represent a good baseline case to evaluate the performance of the GA optimization process. The SRI typical mission profile is presented in Table 6. Table 7 lists the optimization data for this aircraft and Table 8 describes the optimum cycle and flight parameters previously obtained by the author. The SRI drag profile is included in Appendix B.

The discussion about the values found in Table 7 is similar to the GSA case with two exceptions. First, the values of C_1 and C_3 were set at $90\$/lb F_{SL}$ and $600\$/lb W_E$ respectively to represent advanced technology that is not yet as the level of the GSA. Second, all the engine parameters were set at Mattingly's proposed default values.

Table 6. Short Range Interceptor Mission Profile

<u>Leg</u>	<u>Altitude (ft)</u>	<u>Mach Number</u>	<u>Comments</u>
1. Warm Up, Taxi	2000	0	Idle
2. Takeoff	2000	0.2	Takeoff distance: 1,500ft, max power
3. Accelerate to climb speed	0	0.2-0.7	Mil power
4. Climb and accelerate to cruise altitude	0-30,000	0.7-0.9	Mil power
5. Accelerate to supercruise	30,000	0.9-1.5	Mil power
6. supercruise	30,000	1.5	Mil power, distance 250nm
7. Launch AMRAAM	-	-	-
8. 360o, 5g turn	30,000	1.6	Max power
9. 2 x 360° turns	30,000	0.9	Max power
10. Combat acceleration	30,000	0.8-1.6	Max power
11. Fire AIM-9, gun	-	-	-
12. Escape Dash	30,000	1.5	Mil power distance 25nm
13. Return subsonic cruise	30,000	0.9	Part mil power distance 200nm
14. Loiter	10,000	0.41	Part mil power
15. Land	2000	0.2	landing distance: 1,500 ft
Maximum Mach/ maximum altitude leg	50,000	2.5	Max power

Table 7. Short Range Interceptor Optimization Data

<u>General Data</u>	<u>Engine Parameters</u>	<u>Constraints</u>
$K_1 = 1$	$C_{pc} = 0.238 \text{ BTU/lbm-R}$	$T_{t4} \leq 3,200^\circ \text{ R}$
$K_2 = 1$	$C_{pt} = 0.295 \text{ BTU/lbm-R}$	$T_{t7} \leq 3,600^\circ \text{ R}$
$K_3 = 1$	$\gamma_c = 1.4$	$m_0 \leq 270 \text{ lbm/s}$
$C_1 = 90 \text{ \$/lb thrust}$	$\gamma_t = 1.3$	$0.2 \leq M_5 \leq 0.6$
$C_2 = 5,000,000 \text{ \$}$	$h_f = 18,000 \text{ BTU/lbm}$	$\Delta H_T/\theta \leq 32 \text{ BTU/lbm}$
$C_3 = 600\text{\$/lb } W_E$	Cooling air #1 = 0.05	$0.2 \leq M_{5'} \leq 0.95$
$W_{eng}/m_0 = 8$	Cooling air#2 = 0.05	Spools RPM $\leq 110\%$
$m_{0spec} = 37 \text{ lbm/sec-ft}^2 \Pi_B = 0.97$		Cost $\leq 30,000,000 \text{ \$}$
$r_{hub}/r_{tip} = 0.35$	$\Pi_{Dmax} = 0.97$	$r_{tip} \leq 2 \text{ ft}$
$W_{TO} \text{ initial} = 24,800 \text{ lb}$	$\Pi_N = 0.98$	$0.2 \leq \alpha \leq 1.0$
$W_f \text{ initial} = 7,997 \text{ lb}$	$e_{c'} = 0.89$	$3 \leq \Pi_{c'} \leq 5$
$W_P = 2,634 \text{ lb}$	$e_{ch} = 0.9$	$24 \leq \Pi_c \leq 30$
$W_A = 1,000 \text{ lb}$	$e_{tH} = 0.89$	$\Pi_{tL} \leq 0.5$
$W_{PE} = 1,234 \text{ lb}$	$e_{tL} = 0.91$	$P_{t5'} \geq P_{t5}$
$T_{SL}/W_{TO} = 1.14$	$e_{mH} = 0.98$	Takeoff distance $\leq 1,500 \text{ ft}$
$W_{TO}/S = 65 \text{ lb/ft}^2$	$e_{mL} = 0.99$	$P_{t3} \leq 375 \text{ psi}$
No of engines = 1	$e_{mPTO} = 0.98$	$T_{t3} \leq 1,660^\circ \text{ R}$
$rmc = 0.02$	$\eta_{AB} = 0.97$	$F_{SL} \geq 27,523 \text{ lb}$
$M = 0.9$ (material modifier)	$\gamma_{AB} = 1.3$	$W_f \leq 10,000 \text{ lb}$
$a = 2.34$	$C_{PAB} = 0.295$	
$b = -0.13$	$\Pi_{mix} \text{ max} = 0.97$	
$K1$ and C_{D0} = Appendix B	$P_0/P_9 = 1$	
$K2 = 0$	Bleed air = 0.01	
$W_{empty} = 16,803 \text{ lb}$	$\Pi_{AB} = 0.96$	
	$\eta_b = 0.98$	

Table 8. Trial-and-Error Process Short range Interceptor Optimum Design

At design point:			
$M_0 = 1.6$	$h = 35,000 \text{ ft}$	$m_0 = 204 \text{ lbm/s}$	$F_{SL \text{ max}} = 11,199 \text{ lb}$
$\Pi_c = 22.5$	$\Pi_{c'} = 3.7$	$\alpha = 0.6$	$F_{SL \text{ mil}} = 22,365 \text{ lb}$
$T_{t4} = 3,200^\circ \text{ R}$	$T_{t7} = 3,600^\circ \text{ R}$	$M_5 = 0.35$	$C_{TO} = 0.016$
Sea-level performance:			
$M_0 = 0$	$h = 0 \text{ ft}$	$m_0 = 261 \text{ lbm/s}$	$F_{SL \text{ max}} = 30,926 \text{ lb}$
$\Pi_c = 26.3$	$\Pi_{c'} = 4.18$	$\alpha = 0.56$	$F_{SL \text{ mil}} = 19,776 \text{ lb}$
$T_{t4} = 2,968^\circ \text{ R}$	$T_{t7} = 3,600^\circ \text{ R}$	$M_5 = 0.356$	$C_{TO} = 0.016$

III. Methodology

Introduction

This chapter describes in details the objectives of this thesis, the methods used to achieve them, the most significant hurdles that were met during the project, and how those hurdles were circumvented. The on-design and mission analysis optimization processes will also be covered, as are the main assumptions that pertained to the thesis.

Objectives

The overall objective of this thesis was to perform a multidisciplinary and multiobjective constrained optimization of mixed stream, low bypass turbofan design. The optimization approaches had to cover on-design engine optimization at specific flight conditions and engine optimization over a whole mission for a given aircraft.

To achieve the main goal stated above, several sub-objectives had to be met as follows:

1. Include aspects of multidisciplinary optimization with the use of on-design and off-design cycle analysis, which are based on thermodynamic principles, and the inclusion of mission analysis, which involves aspects of aerodynamics such as drag and lift.

2. Apply principles of multiobjective optimization with the minimization of engine size and fuel usage, and overall aircraft cost. The different objectives were to be weighted properly with linear scaling.
3. Develop codes to perform on-design, off-design, and mission analysis for a mixed stream, low-bypass turbofan and adapt them to be part of a global objective function.
4. Develop automated approaches to perform on-design engine optimization and engine optimization over an entire aircraft mission.
5. All codes are to be Matlab m-files to create a package that is readily transferable from one operating system to another and which achieves a high level of modularity.

Assumptions

The methods used to achieve the objectives mentioned above were based on the following assumptions:

1. Both the low and high pressure turbines remain choked for a feasible solution.
2. Components' efficiencies are considered constant.
3. Installation losses are set to a constant 9.1%.
4. The engine exhaust mixer is of constant area, i.e. A_5 / A_5 is constant.

5. A_8/A_{8R} is constant and is equal to one, i.e. the exit nozzle has a fixed throat.
6. Basic aircraft data such as T_{SL}/W_{TO} , W_{TO}/S , drag profiles, and an initial value for W_{TO} are available.
7. The aerodynamic coefficient K_2 is set to zero since both the Global Strike Aircraft (GSA) and the Short Range Interceptor (SRI) are high performance combat aircraft assumed to have uncambered wings (10:39).
8. The reference flight condition for engine optimization over a whole mission is sea-level static.
9. The convergence of W_{TO} as W_f varies is accomplished as part of the genetic algorithm optimization only. It is assumed that fuel weight, and hence W_{TO} , will vary little during the local optimization process. The purpose of this assumption is to ensure that the local optimizer remains well-behaved.
10. High-temperature effects on C_p and C_v are neglected to allow the use as-is of the engine models described in Reference 10. This affects only the GSA since the technology used for this aircraft allows very high turbine temperatures.
11. Other on-design and off-design analysis assumptions are as stated in Chapter 2.

Optimization Problems Statements

This thesis attempted to resolve three different types of optimization problems. The first type was on-design multiobjective engine optimization at a given flight condition.

The second type was multiobjective optimization over a whole mission. For these two types, an objective function of the form shown in Equation (28) below was chosen to depend explicitly on the system parameters of fuel consumption, aircraft weight, and engine inlet annulus area. The third type was single-objective optimization over a whole mission. This last type was a sub-case of the second type. It was used to compare optimization results with the design and performance data previously obtained by the author and to investigate the impact of the aircraft cost and engine annulus area sub-objectives. The problem statement for each type are described below.

Type 1. Multiobjective on-design optimization at a given flight condition.

Maximize:

$$F(\mathbf{x}) = K_1 S + K_2 \text{Cost} + K_3 \text{Area} \quad (28)$$

where

$F(\mathbf{x})$ = global cost function

\mathbf{x} = design variable vector: x_1 = compressor pressure ratio, Π_c

x_2 = fan pressure ratio, Π_f

x_3 = high pressure turbine inlet temperature, T_{14}

x_4 = afterburner temperature, T_{17}

x_5 = bypass ratio, α

x_6 = mass flow, m_0

x_7 = power takeoff fraction, C_{TO}

x_8 = core flow Mach number at mixer, M_5

K_i = individual objective weight factors

sS = linearly scaled specific fuel consumption, as obtained from onx.m and

Equation (3)

$sCost$ = linearly scaled aircraft cost, as obtained from Equations (3), (15), (17),

(18), and data from onx.m

$sArea$ = linearly scaled engine annulus area, as obtained from Equations (3), (21),

and data from onx.m

Subject to:

$T_{t4} \leq \text{maximum}$

$T_{t7} \leq \text{maximum}$

$m_0 \leq \text{maximum}$

$M_5 \geq \text{minimum}$

$M_5 \leq \text{maximum}$

$\Delta h_t/\psi \leq \text{maximum}$

$M_{5^*} \geq \text{minimum}$

$M_{5^*} \leq \text{maximum}$

Spools RPM \leq maximum

Cost \leq maximum

$r_{tip} \leq \text{maximum}$

$\alpha \geq \text{minimum}$

$\alpha \leq \text{maximum}$

$\Pi_c \geq \text{minimum}$

$\Pi_c \leq \text{maximum}$

$\Pi_{c^*} \geq \text{maximum}$

$$\begin{aligned}
\Pi_c' &\leq \text{minimum} \\
\Pi_{IL} &\leq 0.5 \\
P_{t5}' &\geq P_{t5} \\
P_{t3} &\leq \text{maximum} \\
T_{t3} &\leq \text{maximum}
\end{aligned}$$

Type 2. Multiobjective engine optimization over whole mission:

Maximize:

$$F(\mathbf{x}) = K_1 sWf + K_2 sCost + K_3 sArea \quad (29)$$

where

sWf = linearly scaled mission fuel usage as obtained from miss.m and

Equation (3)

Subject to:

Same constraint as Type 1 above
Takeoff distance \leq maximum

Moreover, for a design to be considered feasible, it must accomplish the whole mission by not failing any legs, and the maximum Mach/maximum altitude leg, which is usually not part of the mission profile, must also be flown successfully. The best engine is thus the design that meets all mission requirements with the lowest fuel usage, engine size and aircraft cost.

Type 3. Single-objective engine optimization over whole mission:

Maximize:

$$f(\mathbf{x}) = sWf \quad (30)$$

Subject to:

Same constraints as Type 2 above, except for the cost and engine radius constraints, which are ignored

Type 3 is a sub-case of Type 2 and it is used to compare the results generated by the optimizer with those obtained earlier by the author and to investigate the impact of the aircraft cost and annulus area sub-objectives.

Cases Investigated. Engine optimization was performed to test the validity of the overall optimization processes and their related engine and optimization computer codes on various cases as follows:

Case 1. A simple three-leg mission with the SRI used to test the Type 2 optimization process, since it ran considerably faster than a full fledge mission (hours vs. days). The mission profile is described in details in Chapter 4.

Case 2. Type 1 on-design optimization performed with the SRI at $M_0 = 1.5$ and $h = 30,000$ ft. This case was used for comparison with the design presented in Table 8 in order to investigate the difference between on-design and full-mission optimizations. The flight conditions were selected because the aircraft spends the majority of the mission at or near those conditions. For this case, only the minimization of the specific fuel consumption was considered to allow comparison with the original results presented in Table 8.

Case 3. A Type 3 optimization with the SRI that used a complete 19-leg mission and with a maximum Mach/maximum altitude leg. This was the case used to compare the overall optimization algorithm with data previously obtained by the author using the trial-and-error process mentioned in Chapter 2. Constraints on P_{13} and T_{13} at the design point were ignored to ensure that Case 3 was performed under the same conditions as the original case mentioned above. The value for m_{0spec} was set to 37 lbm/sec-ft^2 for the same reason.

Case 4. A Type 3 optimization performed with the GSA that included the 19-leg mission. This case was compared with Case 5 to investigate the impact of the size and engine annulus area sub-objectives and to investigate the behavior of the GSA gross weight determination model.

Case 5. A type 2 optimization performed with the GSA, fully constrained, and with a 19-leg mission. This case was selected as a case study by Wright Laboratory.

Case 6. Type 1 optimization performed with the GSA at $M_0 = 1.5$ and $h = 60,000$ ft for comparison with Case 5. The reasons to do so are similar to those of Case 2.

The outputs of each case include the design variables values for the best design, all the outputs listed in Table 1, and, if applicable, values for W_{TO} , W_f , aircraft cost and annulus area. The results for these cases are presented and discussed in Chapter 4.

Overall Approach Taken to Achieve Thesis Objectives

The thesis objectives were achieved with the completion of the following steps, which are described in details in later sections:

1. Creation of the engine on-design analysis computer code.
2. Creation of the engine off-design analysis computer code.
3. Creation of the mission analysis computer code.
4. On-design engine design optimization with global genetic algorithm (GA) and local gradient-based optimizer.

5. Full-mission engine design optimization with global GA and local gradient based optimizer.

The main drive of the project was not to create new tools but to use existing methods, such as GAs and engine performance analysis, together in novel ways in order to automate engine conceptual optimization. Hence a major aspect of the work accomplished in this thesis consisted in building interfaces between on-design, off-design and mission analysis codes with both the GA optimizer and the local gradient-based optimizer. This implied that all the computer codes used had to be modified in order to work together seamlessly, with minimal user input, to perform the overall optimization process. The modifications to the codes affected the way data are transferred between programs but not how their respective outputs were determined.

All the codes used and written for this thesis are available and can be obtained by following the instructions included at Appendix A. The instructions on how to set up a problem and perform an optimization run are included in Appendix C.

Matlab as a Programming Environment. As mentioned earlier, it was decided to use Matlab to program all the required codes and perform the optimization. The reasons for this choice are numerous. Matlab is a powerful mathematical application which can efficiently perform large numbers of iterative calculations. Matlab comes with its own simple and user-friendly programming language which does not require compilation to run programs. Moreover, many optimization routines are included as part of the Matlab

optimization toolbox and many more are available from various educational institutions' Web sites.

Matlab executable files all use the '.m' extension and are commonly referred as m-files. These file can be simple batch files or complex mathematical subroutines. The main advantage of the m-files system is its inherent modularity. It allows the substitution of subroutines in a program by the simple change of a word in the program. This modularity is crucial to this thesis as this allows for growth of the overall optimization algorithms. For example, advanced installation losses or cost models could be created as m-files and included in the original objective function file.

Another advantage of Matlab is that the language used is Matlab, and not operating system, specific. This means that any m-file will work without modification on any computer with a functional copy of Matlab. For example, an m-file written on a Unix machine would work just as well on a DOS machine. The only exception to this is that m-files names for DOS machine can be no longer than eight characters long. The codes used in this thesis were designed for a Unix environment and, as a results of this, some m-files names are longer than eight characters. They would thus require some modifications to run under DOS.

Creation of the On-Design Analysis Computer Code

The on-design analysis code created by the author for this thesis is used as part of the on-design optimization objective function and also as part of the off-design and mission analysis. It is based on the theory and approach covered in Chapter 2.

To perform the analysis, the code requires as input a 1×8 array with the value of the eight design variables (Π_c , Π_c' , T_{i4} , T_{i7} , α , m_0 , C_{TO} , and M_5 respectively). The program then initializes the engine parameters (such as efficiencies and flight conditions) and evaluates the altitude properties at the given flight conditions. The program performs on-design calculations and, as the last step, sets the results obtained as reference values to be used by the off-design code. All the engine parameters are included as part of the on-design code. The altitude properties, such as air pressure and temperature, are calculated with an m-file at altitude up to 65,000 ft, using equations developed in Chapter 1 of Reference 2.

The program was made robust to ensure viable answers were provided, even for unfeasible points. This was required for use with the optimization routines, since unworkable values, such as imaginary numbers or division by zero, would make the optimization process fail. Those problems were averted by ensuring P_{15}'/P_{15} and P_{19}/P_9 values were greater than one. This was accomplished by the assignment of values greater than one to those parameters if the case arose where those conditions were not met. A flag inside the program was also set to indicate that a point failing one or more of the above conditions was unfeasible. A similar technique was used to ensure values for M_6 were real numbers. The accuracy of the program was tested using examples and data from Chapter 4 of Reference 10.

The stand-alone on-design code, 'onx.m', required two major modifications to be used as part of the optimization process. First, the program was turned into a Matlab function. A Matlab function is like a Fortran subroutine in that it only accepts specific

inputs and produce outputs. This was necessary to ensure any on-design analysis performed as part of the optimization would not be corrupted by parameters values from earlier function calls or other codes. The second modification involved the inclusion, as part of the on-design code, of the global objective function parameters and linear scaling data.

The stand-alone on-design code is 'onx.m' while the version used as part of the optimizer is 'onxopt.m'.

Creation of the Off-Design Analysis Computer Code

The off-design analysis program was developed by the author and is based on the theory covered in Chapter 2. It is used as a stand-alone program and as an integral part of the mission analysis.

Inputs for the off-design analysis include the design point (in the same 1×8 array used by the on-design program) and a 1×6 array containing off-design conditions (M_0 , h , afterburner setting (on or off), P_0/P_9 , and starting values for T_{t4} and T_{t7} , respectively). The code starts by performing an on-design analysis to obtain the reference point parameters. Altitude properties at the off-design conditions are evaluated in the same manner as the on-design case. The program then performs off-design calculations, with iterations on τ_{tL} , α' and m_0 required to converge toward a solution. A Newtonian iteration technique (10:150) is used to obtain τ_{tL} so the magnitude of the τ_{tL} step size between iterations is based on a difference between a calculated τ_{tL} value and the value obtained from the

previous iteration. Iterations on α' and m_0 are based on a functional iteration scheme where the value of the parameter of interest is set to a calculated value until the convergence criterion is met (10:150). Moreover, as indicated in Figure 4, fixed-step iterations on τ_{iL} are used to keep P_{t5}/P_5 and M_5 within acceptable limits. If those precautions are not taken, the equations that defined those parameters would behave badly and usually fail to converge to a solution. Once component behaviors are determined, overall engine performance at the off-design conditions is evaluated. The iterative scheme used in the author's code is largely based on Mattingly's original ONX and OFFX codes (12).

Another ill-behaved process is the convergence of τ_{iL} , when the difference between the reference and the off-design value for T_{t4} is large, i.e. several hundred degrees. Under those circumstances, τ_{iL} may fail to converge. To remedy this situation, T_{t4} is throttled from the reference value to the off-design in fixed steps. The input value of T_{t4} to the off-design code becomes the 'target value'. The algorithm start with the maximum allowable value for T_{t4} , as defined by the user, and throttles from that maximum value to the target value in small enough steps so as to check and ensure convergence of τ_{iL} . The step size is related to the difference between the present iteration T_{t4} value and the target value. Large step sizes are used when the difference is large and smaller ones are used as T_{t4} converges to its target value.

An additional feature of the off-design code are the limits set on the compressor exit pressure (P_{t3}) and temperature (T_{t3}), and shaft rpm. When upper limits are violated on those parameters, the program iteratively (with the use of a Newtonian scheme similar to

the one used for τ_{LL}) throttles back the off-design T_{t4} until the constraints on these parameters are met. This implies that off-design performance must be re-evaluated at each iteration of T_{t4} .

The off-design analysis code has to be made very robust for the same reasons as those stated for the on-design case. Off-design parameters have to be evaluated, even for infeasible points, in order for the mission analysis code to provide a fuel weight value to the optimizer. Robustness is achieved by the inclusion of the same protections incorporated in the on-design code, namely protection against infeasible values of P_{t5}/P_{t5} , P_{t9}/P_9 , and M_6 . Moreover, if any of the off-design parameters discussed above do not converge within a set number of iterations, the whole off-design iterative sequence is aborted and off-design performance is then evaluated using the most recent values of the components parameters. A flag is then raised to indicate that the given off-design condition is infeasible.

The stand-alone off-design code is 'offx.m' while the version used in the optimization process is 'offxmiss.m'. In the 'offxmiss.m' program, on-design analysis is not required since it is included in the mission analysis code. Penalties are also included in the program at all points where parameters might fail to converge, i.e. M_5 , P_{t5}/P_{t5} , M_5 , τ_{LL} , α' , m_0 , P_{t9}/P_9 , T_{t4} , P_{t3} , T_{t3} , both shaft rpm, and points where the high pressure turbine might be unchoked ($\Pi_{LL} > 0.5$). These penalties are determined with the use of an appropriate form of Equation (9) and were added to the global mission analysis objective function to indicate unfeasible points as discussed in Chapter 2. Since a whole, multi-leg mission may fail more than one leg, the penalties are calculated in such a way that only the

highest value for a given penalty is retained for transmission to the objective function. For example, if three legs fail to achieve m_0 convergence, only the penalty from the leg that failed to converge by the largest margin is retained.

Creation of the Mission Analysis Computer Code

Again, the mission analysis code used in this thesis was developed from the theory covered in Chapter 2. The code is used as a stand alone-program and as part of the mission optimization objective function.

The inputs to the mission analysis code include the same 1×8 array used in the on-design and off-design codes and a mission profile included in an m-file. The mission profile file contains the number of legs, an array describing each leg of the mission, and the maximum Mach/maximum altitude leg data, if applicable. The structure of the mission file is covered in Appendix C.

The program first step is to initialize aircraft data such as W_{TO} , T_{SL}/W_{TO} , W_{TO}/S , and so on. On-design cycle analysis is then performed at sea level static. The data obtained from this analysis will be used to determine reference values and the thrust lapse, T/T_{SL} , for the mission legs off-design conditions. If the design point is not at sea-level static, the code has the option to evaluate the reference data at the desired point and perform an off-design analysis at sea-level static to determine F_{SL} . Once the mission profile is loaded, the initial fuel fraction β is set to one and the mission analysis proper is performed leg by leg.

Mission analysis for each leg begins with the input of the leg off-design conditions from the mission profile. Off-design analysis is then performed to obtain thrust and fuel consumption data and also verify the leg's off-design feasibility for the given baseline engine. Drag parameters K_1 , K_2 and C_{D0} are then determined from profiles included in m-files. The drag profiles for the SRI and the GSA are included in Appendix B. The drag parameters for the SRI are a function of simple linear equations while the GSA drag data is estimated with a third degree polynomial curve fit from data obtained in Reference 4.

From the drag data and flight conditions, aerodynamic parameters such as C_D , drag and C_L are evaluated. Legs where $P_s = 0$, such as cruise, loiter and constant speed turns, require the reduction of the available installed thrust until it matches the required installed thrust. In those instances, the required thrust equals the drag. The thrust reduction is achieved by iteratively throttling back the off-design T_{17} (if applicable) and T_{14} with the Newtonian scheme discussed before until the desired thrust is achieved. If it is not possible to throttle to the desired thrust, an appropriate penalty is applied to the global mission objective function based on Equation (9) and a flag is raised to indicate an infeasible solution. The reduction of the available thrust to the required thrust is essential to ensure that fuel usage is kept to a minimum and to verify that a given engine design can throttle back far enough to meet all mission requirements. If it is determined that there is not enough thrust for a given leg, an appropriate penalty is applied and a flag is raised to indicate an infeasible solution. The weight fraction Π_f and the fuel fraction β for the leg is then calculated (10:209). In cases where the values of β are too low and diverge to zero over the mission, β is given a set value and a penalty is applied.

Once all the mission legs have been analyzed, the fuel weight, W_f , for the whole mission is determined as are the takeoff distance and the feasibility of the maximum Mach/maximum altitude leg (if applicable). Again, in cases where the takeoff distance constraint is not met, an appropriate penalty is applied and a flag is raised. The feasibility of the maximum Mach/maximum altitude leg is evaluated with the off-design code. If the point is infeasible for any reason a flag is raised that indicate that this point, and hence the whole mission, failed. A penalty is applied as described in the off-design code section. Finally, W_{TO} is re-evaluated with Equations (10), (11), (12) for the SRI, or Equation (13) for the GSA, and the methods covered in Chapter 2.

The mission analysis program is inherently robust since all the major protections against design failure are included as part of the on-design and off-design codes. The code evaluates W_f and the penalties to be included as part the objective function if the design fails the mission. The application of penalties as part of the mission analysis is of particular importance to the optimization process. The behavior of W_f when a mission failed was unpredictable and the fuel weight could jump up or down by as much as a 1,000 lb (for the SRI) between infeasible points. The unpredictable nature of W_f depended mainly on which part the off-design analysis did not converge for unfeasible points. However, it was observed these variations in W_f had little impact on the global objective function values since any irregularities in the behavior of W_f due to unfeasible points were drowned out by the size of the total penalty, which was typically an order of magnitude larger than the global objective function value.

The mission analysis code was validated using the data and mission profile from the original SRI data obtained by the author. The process was slow, especially for legs that required throttling back the thrust, such as low speed cruise and loiter legs. A typical full mission, such as the SRI 19-leg mission, would require between one and five minutes to find a solution. The main reason for this slow response is the large number of off-design performance evaluations required by the code. This number could range in the hundreds for missions with legs that require throttling back the thrust, such as low speed cruise.

The stand-alone mission analysis program used in this thesis is 'miss.m' while the ones used as part of the optimization process are 'gamiss.m' and 'gamiss2.m'. 'gamiss.m' is used as part of the global optimization while 'gamiss2.m' is tailored for use with the local optimizer. The main differences between these programs is that 'miss.m' does not evaluate penalties while 'gamiss.m' and 'gamiss2.m' have no provision to evaluate W_{TO} , since this calculation is integral to the genetic algorithm as explained below. Also, 'gamiss.m' and 'gamiss2.m' are set as Matlab functions to avoid the corruption of a given mission analysis with data from earlier iterations.

On-Design Optimization Process

The on-design engine design optimization process at a given flight condition, as performed in this thesis, involves the use of the on-design analysis code with a genetic

algorithm global optimizer and a sequential quadratic programming (SQP) gradient-based local optimizer. On-design optimization is normally of limited usefulness since an aircraft rarely flies at the design condition. The reasons this kind of problem was included as part of the project are threefold. First, since on-design analysis is practically instantaneous compared to a mission analysis, it was a useful tool for testing programs such as the GA code and concepts like penalty functions. Second, when performed with a GA, on-design optimization proved to be a good way to test the robustness of the on-design code since the GA tries design points all over the design space, or to estimate proper bounds for design variables. Third, on-design optimization might prove useful in the design of a weapon system such as a cruise missile, which operates at a given altitude and speed for most of its flight.

It was decided to use a two-step process to perform the optimization since the size and shape of the feasible engine design region are difficult to evaluate. This is due to the complex interactions of the design variables in the determination of on-design calculations. The first step of the optimization process would consist of a search of the entire design space with a global optimizer. The purpose of this search would be to identify good feasible points and improve on them to isolate the best candidates for optimum designs. The second step of the process consist of a local optimization, performed by a gradient-based optimization algorithm, with the best designs provided by the global optimizer as starting points. The purpose of the local optimization is to converge rapidly to the optimal solution, as gradient-based algorithms are very efficient and converge to a local minima

quickly. A genetic algorithm (GA) was selected to perform the global optimization because a GA is an excellent tool to explore design spaces and locate regions where global minimas or maximas exist.

The on-design optimization process is controlled by a master program, in this case, an m-file. This process is illustrated in Figure 7 and is included in the file 'gaondes.m'. The inputs required for the optimization program are the design variables boundaries, and the name of the objective function file. Since the sequential quadratic programming (SQP) optimizer, 'constr.m', uses matrices to compute and store gradient information, the values of the design variables for each design are scaled to the same order of magnitude. This is necessary in order to avoid Matlab errors due to badly scaled matrices. For example, scaling problem might arise due to the fact that C_{TO} is of the order of 0.01 and T_{14} is expressed in thousands of degrees. The design variables are scaled back to their proper value as part of the objective function value determination. Additional inputs include the maximum number of generations and the name of the termination function, if different than the default parameters. A user defined termination function allows the optimization to be stopped by a different criterion than the default maximum number of generations. The default was used for this case.

Once these inputs are received, the genetic algorithm and the objective function are called and the initial random population is created. The GA used is GAOT, as described in Chapter 2. The initial population size and the number of generations were left at their default values of 80 and 50 respectively. This proved more than adequate to obtain good

feasible solutions. The best design, as determined by the GA was fed to the SQP local optimizer, the Matlab's 'constr.m' code described in Chapter 2. Once the local optimizer has converged to the optimum design, on-design analysis is performed to obtain the performance data for this point.

Since the GA handles only maximization problems, it was well suited for the problems at hand, which require the maximization of the linearly scaled objectives for fuel usage, cost and engine area. However, the value of the local objective function had to be preceded by a minus sign since the gradient-based optimizer minimizes the objective function. Moreover, a useful characteristic of GAOT is the variable population size. As poor designs are weeded out, the population decreases over time and includes only the best individuals. This has the advantages of speeding up the process because it considerably reduces the required number of designs to evaluate.

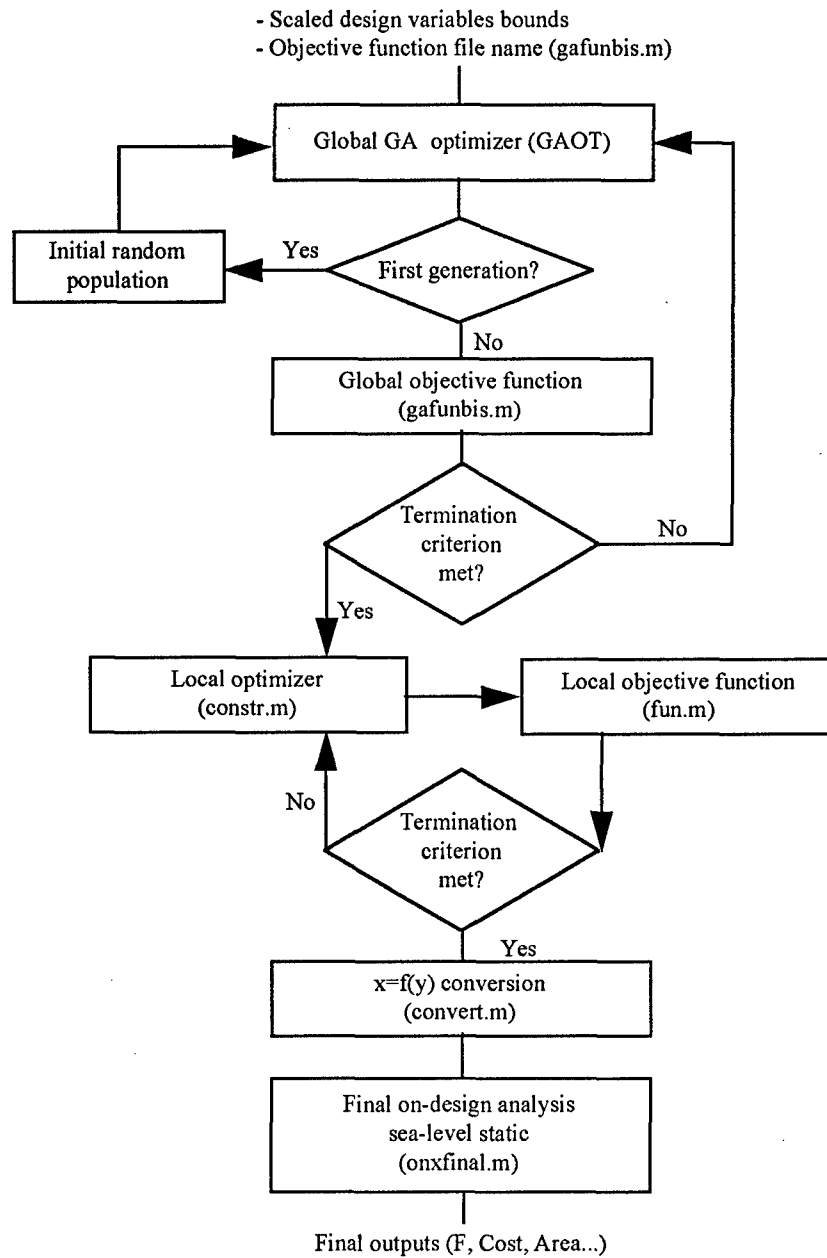


Figure 7. On-Design Optimization Process (gaondes.m)

On-Design Objective Function. Since the objective function file evaluates not only the function value but also constraints information, it requires further elaboration. Every time the GA optimizer calls the function file, several operations are performed. The objective function evaluation process is illustrated in Figure 8 and is included in the file 'gafunbis.m'. When the GA transmits a design to the function file, the design values are converted to their proper values, as described above. The next step involves the performance of the on-design analysis. With the data obtained from this analysis, the value of the different terms are calculated and transformed with linear scaling. The constraint functions are then evaluated and penalties are applied for each constraint that is not met. Penalties for each constraint are determined with the appropriate form of Equation (9). The global objective function value is obtained by the addition of all the individual objectives terms and the subtraction of all the penalties.

The objective function process used as part of the local optimization is similar to the global case and is illustrated in Figure 9. The main difference resides in the fact that since the SQP algorithm deals with constraints directly, there is no need for penalty functions. The on-design optimization local objective function is included in the file 'fun.m'.

Detailed instructions on how to perform on-design engine optimization with the codes developed for this thesis are included in Appendix C. Although no built-in interface was created to set-up a case, doing so only requires the modification of three files. The

variables boundaries are set with 'gafunmiss.m', the engine parameters are set in 'onxopt.m' and the constraints' limits are set with 'gafunbis.m'

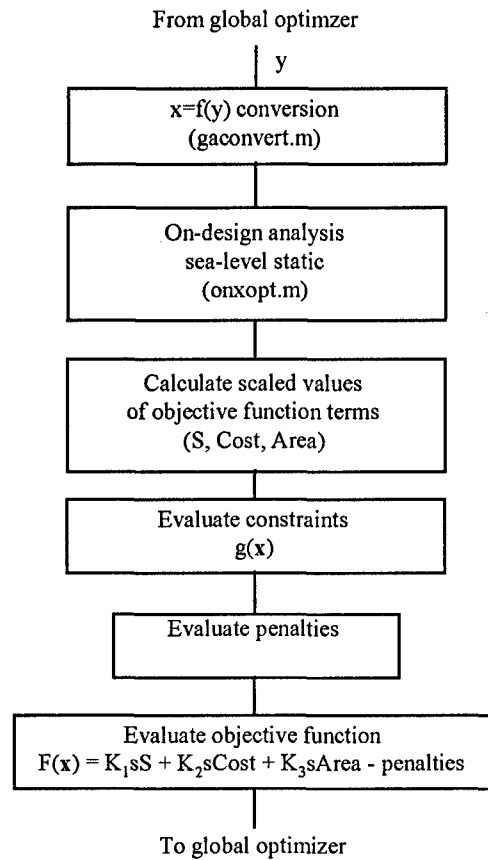


Figure 8. On-design Global Optimization Objective Function Process (gafunbis.m)

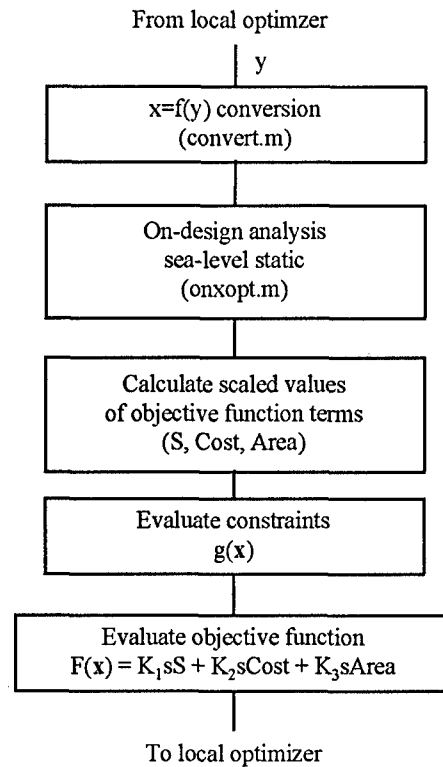


Figure 9. On-Design Local Optimization Objective Function Process (fun.m)

Optimization with Mission Process

Engine design optimization over a whole mission involves the use of the on-design, the off-design and the mission analysis codes. The latter is at the heart of the objective function while the former are components of the mission analysis. As with on-design analysis, and for the same reasons, a global optimization is performed with a genetic algorithm to be followed by a local optimization with a gradient-based algorithm on the mission process.

The master file that controls the mission optimization is 'gaoptmiss.m'. The overall process is illustrated in Figure 10 and 11. The inputs to the optimizer are the same as the on-design case, i.e. scaled values of the design variable and the objective function file name. Again, the default termination criterion was used. For this type of optimization, it was decided to use the default 50 generations for the short range interceptor and between 25 and 30 generations for the Global Strike Aircraft. The initial population was set to 100 individual designs. The larger initial population was selected because the heavy mission and performance requirements usually associated with combat aircraft tend to severely restrict the feasible design space. A larger population increases the probability of finding a point in, or at least near, the feasible region. With the inputs provided, the GA performs the same operations as per the on-design case, with one major exception. The genetic algorithm for mission optimization used was a version of GAOT modified to converge toward a final value for aircraft gross weight, W_{TO} . The procedure selected to update W_{TO} is described in a later part of this section.

Once the GA has operated through the required number of generations, it sends the three best feasible designs to the local optimizer. Three best points were selected, instead of only the best design, to ensure that promising solutions were explored. Another factor is that a good point discovered late in the optimization process might not have been fully investigated by the GA by the time the termination criterion is met. The selection of the three best points increases the probability of finding the best solution. If no feasible solutions are found, local optimization is not performed and the optimization process

stops. This measure is necessary because of the numerous and unpredictable ways a given design may fail a mission, discussed earlier in this chapter. Because it cannot predict what variables to adjust (and how to adjust them) to meet the mission constraint, a gradient-based algorithm would fail to converge to a solution since the gradient information it produces could not be relied upon.

The local optimizer used for mission optimization is Matlab's 'constr.m'. It has been modified to accept W_{TO} as an input but it does not perform W_{TO} convergence. W_{TO} is not updated as part of the local optimization because W_f , and hence W_{TO} , should not vary significantly since it is assumed that the point provided by the GA is near the local minima. As a protection against a design becoming infeasible as it is improved by the optimizer, the value of W_f was multiplied by 1.5 if the mission failed. This encouraged the local optimizer to stay away from the infeasible mission space.

The last step of the mission optimization is to obtain performance results for all three points with final calls to the on-design and mission analysis codes.

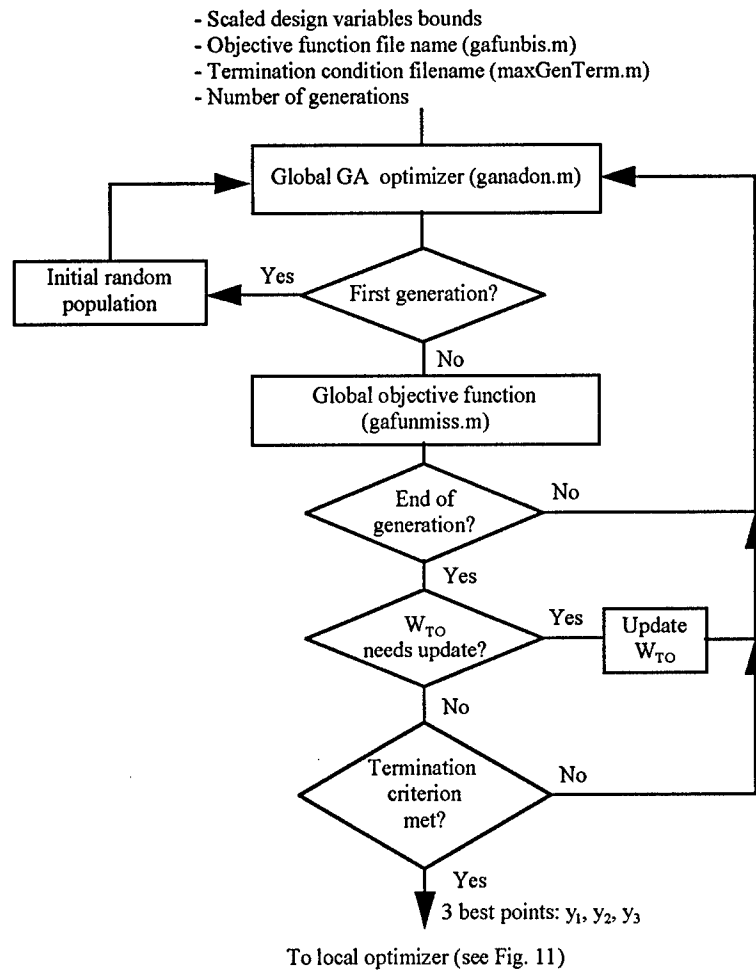


Figure 10. Engine Optimization with Mission Process (gaoptmiss.m) - Part One

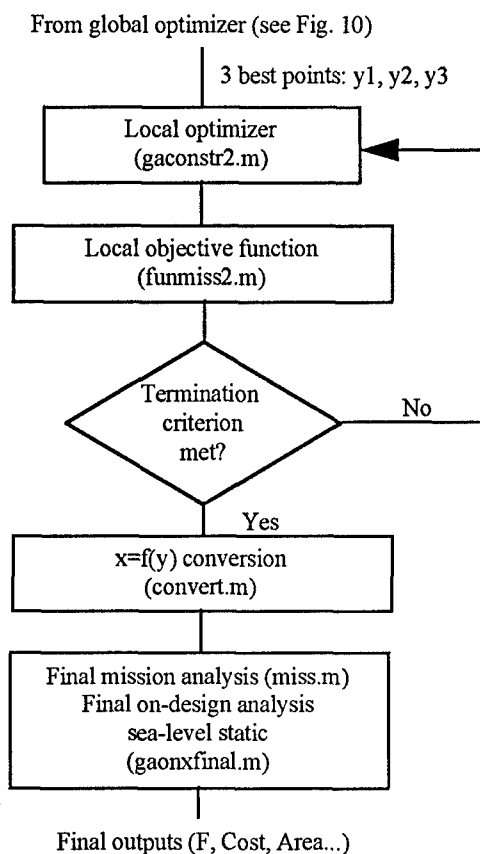


Figure 11. Engine Optimization with Mission Process (gaoptmiss.m) - Part Two

W_{TO} Convergence. The update of W_{TO} is essential to the evaluation of the cost term of the objective function, Equation (29). Equation (10) clearly indicates that W_{TO} is a function of the mission fuel weight, W_f . W_{TO} decreases when W_f decreases and vice-versa. Because W_f is coupled to W_{TO} , it can be shown from Equation (17), the aircraft empty weight (W_E) equation, that if W_{TO} is fixed and W_f varies, the value of W_E would be meaningless. For example, if W_{TO} is fixed and W_f decreases (a desirable result), the value of W_E would increase, as indicated by Equation (17), and hence the size and cost of the

aircraft would also increase, which is the opposite of the expected behavior for W_{TO} . In fact, this situation might force the optimizer to diverge away from the optimal design as it could force the optimizer to maximize fuel weight in order to minimize cost. The update of W_{TO} as W_f varies corrects this irregularity.

The determination of a method to allow the convergence of W_{TO} toward its final value proved to be a major hurdle since the variation of W_{TO} from one generation to the next might affect the fitness of an earlier design and thus induce instabilities in the GA behavior. Fortunately, it was observed that once the algorithm converges toward a solution, W_{TO} would also converge. It was decided for all cases to update W_{TO} at the end of every generation with the W_f value of the best feasible design of a generation. At the beginning of an optimization run, W_{TO} is set to a realistic initial estimate as obtained by the airframe designers. If a feasible point is found for the generation, a new value of W_{TO} is determined from Equations (10), (11), and (12) for the SRI or Equation (13) for the GSA. In order to avoid the corruption of future generations with points whose performance were determined with different values of W_{TO} , the value of W_f for the best point in a generation was reset to an initial value at every generation until the value of W_{TO} met the convergence criteria. If no feasible point were found, W_{TO} was not updated since the value of W_f would be meaningless.

Another problem area was the behavior of W_{TO} when W_f was updated. When W_f decreased, W_{TO} would tend to diverge toward zero, and in cases where W_f increased, W_{TO} would tend to diverge into the unfeasible mission region. In order to control the unstable

behavior of the gross weight, constraints on minimum thrust and maximum allowable fuel weight were applied. These constraints effectively set limits for W_{TO} values. Moreover, to ensure that W_f would not become too high a portion of the gross weight, an additional constraint was applied as follows:

$$W_{TO} - W_f \geq W_{min} \quad (31)$$

where

W_{min} = minimum aircraft weight without fuel (lb)

W_{min} is further defined as:

$$W_{fmin} = W_S + W_A + W_P + W_{Sys} \quad (32)$$

where

W_S = structural weight (lb)

W_A = avionics weight (lb)

W_P = Payload weight (lb)

W_{Sys} = other systems weight (lb)

As for other constraints, if the condition of Equation (31) is not met, an appropriate penalty is applied. At the end of a generation, a convergence scheme to update W_{TO} in decreasing steps was set up for cases where W_{TO} tends to converge toward the minimum allowable W_{TO} value. This step was necessary to ensure That W_{TO} would not be decreased to a value where Equation (31) could no longer be satisfied.

If it is intended to explore potential reductions in structural weight achieved by more efficient engines, the value of W_{\min} provided to the optimization code can be set to a value lower than the value calculated with Equation (32). This would allow the optimizer to, hopefully, further reduce W_f and hence, the gross weight.

There is also a possibility, in situations where W_{TO} increases or with multiobjective optimization, that the best design might have occurred at a value of W_{TO} different from the final weight. The impact of a similar situations on the mono-objective cases is reduced, since the optimum solution tends to be the feasible design with the lowest fuel weight, regardless of W_{TO} . On the other hand, even for mono-objective optimization, the value of W_{TO} becomes a factor with heavily constrained missions when variations of W_{TO} might turn a feasible design into an infeasible one. In order to account for this possibility, the global optimization codes ensure that the value of W_{TO} provided to the local optimizer is the gross weight that was used to determine the best design.

To alleviate some of the limitations and problems mentioned above, another W_{TO} convergence scheme, based on Equation (14) and (15) and included as part of the global

objective function file, was investigated with the full mission SRI case. An appropriate penalty was imposed when the requirement of Equation (15) was not met. As for the other design variables, the side constraints for the dummy fuel weight variable, W_{fdes} , must be set carefully from previously available W_f estimates. This scheme presents numerous advantages. It ensure that each design has an individual W_{TO} value assigned to it, thus eliminating the need to update W_{TO} every generation. This has the potential to considerably reduce the number of generations required since there is no requirement to converge W_{TO} over a run because its design is assigned its own value of the gross weight. Also, there is no need to determine a W_{TO} vs. W_f relationship for a given aircraft since the only aircraft specific data required is the takeoff weight without fuel. Finally, the problem with the unstable behavior of Equation (17), as described above, is eliminated since W_{TO} is always properly matched with W_f .

It is still possible to perform engine optimization for a fixed W_{TO} , if there is a need to solve such a problem. An example of such a situation would be to investigate how the payload or range of an existing aircraft could be increased by the selection of an engine design that provides the lowest fuel usage, W_f . The solution of such a case requires the removal of the W_{TO} update sequence from the GA and replacing the aircraft cost term of the objective function (Equation (18)) with only the engine cost $C_1 F_{SL} N_{eng}$. The procedure to set up a problem where W_{TO} is constant is described in Appendix C.

Mission Optimization Objective Function. Objective function value determination is governed by the file 'gafunmiss.m' and the process is illustrated in Figure 12. The function evaluation procedure starts with the conversion of the scaled design variable to their proper values. The mission analysis function is called upon and evaluates W_f and the feasibility of the mission for the given design. With the data obtained from the analysis, the linearly scaled values of the global objective function terms are evaluated. The constraints are evaluated next and penalties are applied as required.

The penalties for the constraints directly applicable to the design point (i.e. the constraints listed as part of Equation (28)) are applied in the same way as for on-design optimization except for the coefficient at the front of Equation (9), which was set to 1.5 instead of 2. This was done in order to give larger penalties to designs that failed the mission because the mission constraint was considered more critical and difficult to meet. Since a mission is made of several legs, each of which may fail in several ways, it is easier for the optimizer to adjust an infeasible point that passes the mission than to do the opposite. Penalties that arose from a failed mission are calculated with the method described in the mission analysis section above. Once all the penalties are determined, the global objective function value is determined and the three best points are updated if necessary.

The local optimization objective function process is controlled by the file 'funmiss2.m' and it is illustrated in Figure 13. The procedure is similar to the global case.

As for the on-design case, the main difference resides in the fact that since the SQP algorithm deals with constraints directly, there are no need for penalty functions.

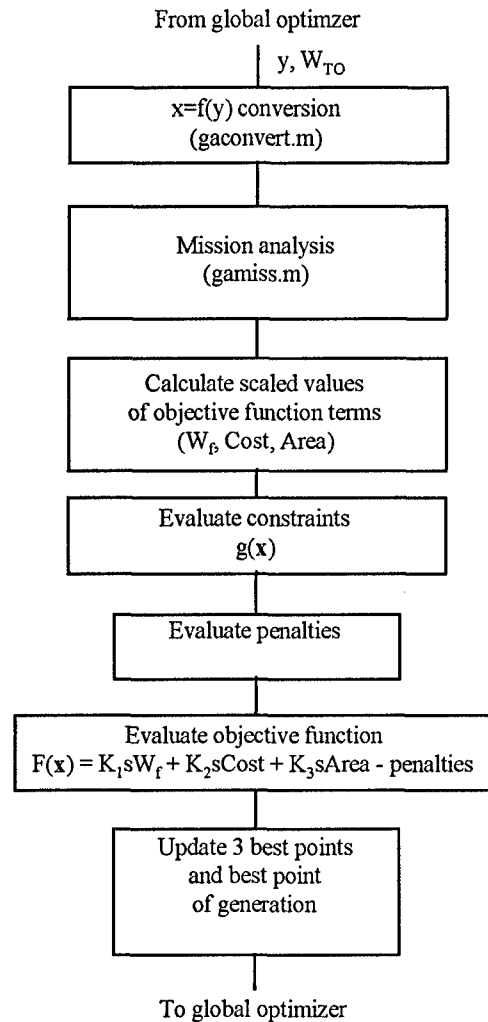


Figure 12. Global Mission Optimization Objective Function Process (gafunmiss.m)

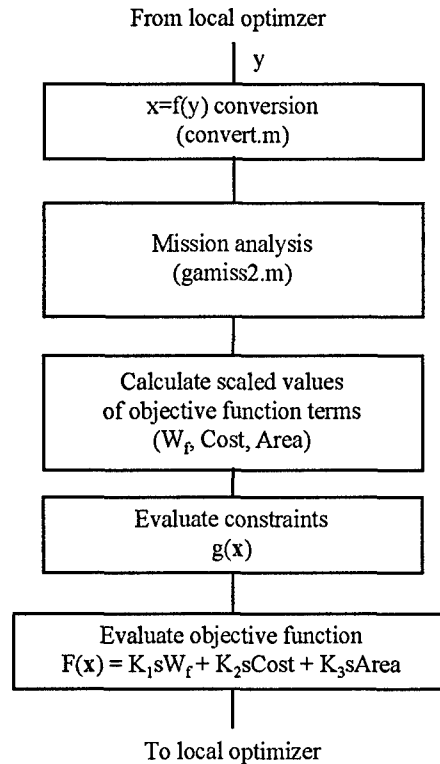


Figure 13. Local Mission Optimization Objective Function Process (funmiss2.m)

Engine design optimization with mission proved to be a slow process. A typical optimization run would take around two to three days for a full mission. It was observed, however, that a good way to make the optimization process as time-effective as possible was to properly bound the design variables with the use of side constraints. With experience, a designer can set limits that will reduce the number of highly unfeasible points, points that often take longer to solve. One has to be careful not to overconstrain the design variables to the point where good feasible solutions are excluded from the design space.

Another option available to speed up the process is to replace the initial random population with population selected by the user. The population thus created must include the objective function values as well as the design variables values. The advantage of such a method is to give the GA a population of feasible and near feasible points that would significantly increase the probability of finding the optimum design. However, this approach is not recommended since the creation of such a population is itself a time-consuming process and it defeats the purpose of using a fully automated tool to accomplish such a search.

Detailed instructions on how to perform engine optimization with mission using the codes developed for this thesis are included in Appendix C. The set up of a case requires the modification of four files. The variables boundaries are set with 'gaoptmiss.m', the engine parameters are set in 'onxopt.m', the mission and aircraft parameters are set in 'gamiss.m' and the constraints' limits are set with 'gafunmiss.m'. A file containing the drag profile for the given aircraft must also be provided.

IV. Findings and Analysis

Introduction

This chapter presents and analyses the results for each of the cases introduced in Chapter 3. The emphasis is on the behavior of the optimization processes and the related computer codes.

General Comments and Observations

The overall processes and computer codes for both on-design and mission optimization proved successful, subject to certain limitations discussed below. The thesis demonstrated the feasibility of automating engine design optimization.

The engine cycle analysis codes proved very robust and provided acceptable data to the optimizers in all cases. The genetic algorithm was able to explore the design space and find feasible designs in five of the six cases investigated. The local optimizer, successfully converged toward the optimum design when provided feasible points from the genetic algorithm, although it failed to do so for unfeasible points or near unfeasible points in two of the six cases.

The six cases investigated were as follows:

Case 1: Test Case with Short Range Interceptor

Case 2: Short Range Interceptor On-design Mono-Objective Optimization

Case 3: Short Range Interceptor Mono-Objective Mission Optimization

Case 4: Global Strike Aircraft Mono-Objective Mission Optimization

Case 5: Global Strike Aircraft Multiobjective Mission Optimization

Case 6: Global Strike Aircraft Multiobjective On-Design Optimization

The process for engine optimization with a mission was slow. In these cases, it took between two and three days of continuous running on a Sun Sparc10 workstation to obtain a solution. The main factor that contributed to the lengthy run duration was the throttling required for cruise and loiter legs. On-design optimization was significantly quicker, with solutions obtained in less than 30 minutes. The on-design flight conditions for all mission analysis cases was sea-level static.

It was found that a good way to speed up the codes and to increase the probability of early entry into feasible space was to properly bound the design variables with the use of side constraints. The upper and lower boundaries for each variable had to be close enough to reject the obvious unfeasible points while not being so restrictive as to also reject good solutions.

Experience and data from on-design and mission optimization runs were used to gauge variable boundaries. Most side constraints for the design variables were set arbitrarily, with design variable ranges estimates based on current engine technology and performance (for the Short Range Interceptor) or on expected future trends (for the Global Strike Aircraft). Moreover, hard technical limitations might also impose limits to the range of values available for a given variable. An example of such technological

constraint is the value maximum turbine inlet temperature, T_{t4} , which is dependent on the properties of the turbine material.

Also of critical importance to ensure the optimizers behaved as predicted was to carefully set the linear scaling boundaries. If a term of the objective function strayed beyond improperly set scaling limits, it was observed that the scaled value of the objective function would undergo a sign change, thus causing the optimizers to diverge and fail. Scaling limits were imposed very conservatively, again using data from experience and optimization runs.

The genetic algorithm found feasible points within three generations for all on-design or mission optimization cases except for the highly constrained short range interceptor full mission case, for which no feasible point was found. As explained in the section that covers that case, it is believed that no feasible point existed for this case due to the engine analysis codes used to solve it. The full mission cases tend to be more constrained, compared to the on-design cases, due to the numerous mission requirements.

The scheme to select the three best points from the global optimizer for local optimization proved to be of no value. The solutions provided by the global optimizer were so similar to each other that they would converge to the same value at the end of the local optimization. Moreover, in cases where W_{TO} had not converged to a final value, only one or two points would be provided by the global optimizer. The same would apply for situations where the global optimizer is near the optimum design and improvement to the objective function value are sporadic.

The techniques used to update the gross weight (W_{TO}) based on Equations (10), (11), (12) and (13), as described in Chapter 3, proved effective, especially in the case of the Global Strike Aircraft cases. Although the weight did not converge by the end of a GSA global optimization run, because the cases were not run for enough generations, significant reductions in W_{TO} were realized. However, these techniques proved sensitive and unstable in situations when the initial W_{TO} must be updated upward, in which cases W_{TO} would diverge into the unfeasible mission region. This divergence was due to the behavior of the global optimizer, which based its best design points on data from earlier generations; data that were obtained on W_{TO} values not properly matched to the fuel weight. Such a situation was encountered in the SRI full mission case. To ensure that W_{TO} converge properly, it was found that the initial gross weight should be overestimated in order to ensure it will be updated downward.

Moreover, when W_{TO} is updated, many good points from earlier generations might become infeasible and this could slow down the optimization convergence. In such a situation, the SRI test mission usually took between one and five generations to return to the feasible region and update W_{TO} . It was also observed that the convergence of W_{TO} toward its final value would take the form of a damp oscillatory motion in the case of the Short Range Interceptor test case. If W_{TO} did not converge by the end of a run it is possible that there might still be room to improve the best design at the end of the global optimization. The remedy to ensure W_{TO} convergence is to increase the number of generations, at the cost of longer runs. A number of generations between 75 and 100 would seem adequate.

The third method to determine W_{TO} , based on Equations (14), (15), and the dummy design variable $x_9 = W_{fdes}$, proved extremely effective in the case of the SRI full mission optimization. Since an individual gross weight is assigned to each design investigated by the global optimizer, there is no need to update W_{TO} at the end of a generation. The main advantages of this method are shorter runs, a value of W_{TO} properly matched with a value of W_f when evaluating aircraft cost (Equation (17)), and the insurance that W_{TO} will not diverge to the unfeasible mission region. Moreover, no complex W_{TO} vs. W_f relationship need be determined for a given aircraft since the only data aircraft specific data required to use the model is the takeoff weight without fuel. It is felt that this model is superior to the schemes based on Equations (10) to (13).

In cases where the gross weight convergence schemes based on Equations (10) to (13) were used the cost term of the objective function is of limited validity if W_{TO} fails to converge. As explained in Chapter 3, viable aircraft cost values depends on W_{TO} varying in unison with the fuel weight, W_f . Moreover, the aircraft cost term of the objective function might get worse once the gross weight has converged and is fixed, since lower fuel weight values increase the aircraft cost as indicated by Equation (17) and its impact on Equation (16). Although this effect has been observed to be minimal when compared to the overall benefit of a lower W_f , it is felt that this discrepancy has to be eliminated with a better behaved cost model. On the other hand, as discussed above, this situation can be alleviated with the use of a gross weight determination model based on Equations (14) and (15).

Three major constraints drove the selection of the best design by the global optimizer. The first challenge for the genetic algorithm was to find designs with enough thrust for all mission legs. The second major constraint to meet involved keeping the high pressure turbine choked. The third hurdle was to ensure the convergence of τ_{LL} off-design. The last two constraints were a factor mainly for loiter and maximum Mach/maximum altitude legs.

In both on-design and mission optimization, the design space appeared to be a flat region pockmarked with small minimas and maximas. This conclusion was reached with the analysis of numerous run results. It was found that many designs that varied in significant ways would give similar values of the objective function. This observation validates the use of genetic algorithm to search the design space due to the presence of multiple minimas. However, it raises the possibility of the GA not finding the global minimum. The only way to ensure the global minimum has been located is by running cases with a larger populations for more generations.

For the cases investigated, fuel consumption was the factor having the greatest impact, by far, on the final design. Engine annulus area turned out to be a minor factor while aircraft cost had negligible impact on the final design. These conclusions are based on the results presented in Tables 16 and 18 that show that for the GSA, W_f varied by only 0.21% between the mono-objective and the multiobjective mission optimization cases ($W_f = 110,179$ lb vs. $W_f = 110,413$ lb respectively) while area varied by 1.63% (from 10.41 ft^2 to 10.24 ft^2). The aircraft cost only decreased by 0.025% (from \$97,808,200 to \$97,782,840). The aircraft cost value tended to be low compared to available data (for

the GSA) or existing similar aircraft (for the SRI). This was attributed to the crudeness of the cost model, which cannot account for factors such as research and development. Further investigation is required to establish the individual impacts of aircraft cost and engine annulus area in more details.

Case 1: Test Case with Short Range Interceptor

The multiobjective 3-leg mission SRI test case was successful and proceeded without interruption. The design variables and scaling parameters boundaries are listed in Table 9. The design and scaling variables side constraints values were based on knowledge previously gained by the author while performing the SRI trial-and-error engine optimization. The results and performance for the optimum design are presented in Table 10. It took about six hours for the optimization codes to converge to a solution. The mission included two supercruise legs and a 5g turn at $M = 0.9$. It did not include maximum Mach/maximum altitude leg. Feasible points were found within the first generation and were improved upon in the next 49 generations. This result was to be expected as the test case was less restricted than cases with full-mission. The local optimizer successfully converged to a solution starting from the best solution provided by the global optimizer.

Table 9. Variables and Linear Scaling Boundaries for Case 1: SRI Test Case

<u>Design Variables Boundaries:</u>			
$24 \leq \Pi_c \leq 30$	$2.5 \leq \Pi_{c'} \leq 5$	$2,900^\circ \text{ R} \leq T_{t4} \leq 3,200^\circ \text{ R}$	$3,200^\circ \text{ R} \leq T_{t7} \leq 3,600^\circ \text{ R}$
$0.2 \leq \alpha \leq 1$	$250 \text{ lb/s} \leq m_0 \leq 270 \text{ lb/s}$	$0.01 \leq C_{TO} \leq 0.02$	$0.2 \leq M_5 \leq 0.6$
<u>Linear Scaling Boundaries:</u>			
$4,000 \text{ lb} \leq W_f \leq 18,000 \text{ lb}$	$\$10,000,000 \leq \text{Cost} \leq \$30,000,000$	$3.14 \text{ ft}^2 \leq \text{Area} \leq 12.56 \text{ ft}^2$	

The gross weight, W_{TO} , successfully converged to a final value. The initial W_{TO} for this case was set to 24,000 lb (vs. 24,800 lb for the final SRI W_{TO} previously obtained by the author) in order to account for the shorter mission which would require less fuel. The final value for W_{TO} was 22,394 lb, which supported the assumption. It should be noted that W_{TO} was updated only every third or fourth generation and did not converge to its final value until generation 46. This suggested that the genetic algorithm was able to improve the best designs throughout the whole run in order to lower the fuel weight and hence, the gross weight.

Table 10. Final Results for Case 1: SRI Test Case

<u>Best point from global optimizer:</u>				
$\Pi_c = 25.4$	$\Pi_{c'} = 3.16$	$T_{t4} = 2,900^\circ \text{ R}$	$T_{t7} = 3,539.5^\circ \text{ R}$	$\alpha = 0.73$
$m_0 = 250.22 \text{ lb/s}$	$C_{TO} = 0.018$	$M_5 = 0.205$	Function = 2.0375	
<u>Second best point from global optimizer:</u>				
$\Pi_c = 25.4$	$\Pi_{c'} = 3.16$	$T_{t4} = 2,900^\circ \text{ R}$	$T_{t7} = 3,539.5^\circ \text{ R}$	$\alpha = 0.73$
$m_0 = 250.22 \text{ lb/s}$	$C_{TO} = 0.018$	$M_5 = 0.205$	Function = 2.0375	
<u>Third best point from global optimizer:</u>				
$\Pi_c = 0$	$\Pi_{c'} = 0$	$T_{t4} = 0$	$T_{t7} = 0$	$\alpha = 0$
$m_0 = 0$	$C_{TO} = 0$	$M_5 = 0$	Function = N/A	
<u>Final design from local optimizer:</u>				
$\Pi_c = 24.65$	$\Pi_{c'} = 2.92$	$T_{t4} = 2,900^\circ \text{ R}$	$T_{t7} = 3,519^\circ \text{ R}$	$\alpha = 1$
$m_0 = 250.22 \text{ lb/s}$	$C_{TO} = 0.01$	$M_5 = 0.306$	Function = 2.052	
<u>Final performance (sea-level static):</u>				
$F_{SLmax} = 26,872 \text{ lb}$	$F_{SLmil} = 15,769 \text{ lb}$	$S_{max} = 1.857 \text{ (1/hr)}$	$S_{mil} = 0.776$	
$F/m_{0max} = 107.49 \text{ (lbf/lbm/s)}$		$F/m_{0mil} = 63.07 \text{ (lbf/lbm/s)}$		
$\Pi_{tL} = 0.413$	$\Pi_{cH} = 7.81$	$\Pi_{tH} = 0.3191$	$P_{TO} = 328,670 \text{ W}$	
Cost = \$13,501,000	Area = 6.76 ft ²	$W_{TO} = 22,394 \text{ lb}$	$W_f = 6,622 \text{ lb}$	

As indicated in Table 10, the global optimizer provided only one point to the optimizer, and not the three best designs, as would be expected. It should also be noted that the second best point is identical to the first. This is explained by the fact that W_{TO} converged late in the run, thus not leaving enough generations to improve on the best design. The fact that the optimum design did not improve in the last four generations seemed to indicate that the global optimizer had converged near the global minimum and that it was appropriate at this point to switch to the local optimizer.

Based on experience, the results in Table 10 appear realistic. It was also verified that all the constraints were met. The local optimizer increased the scaled global objective function value from 2.0375 to 2.052, which represented a 0.71% improvement. Since the value of fuel weight W_f was linearly scaled and W_{TO} was not updated for the local optimization, it is assumed that the improvement in W_f is of the same order of magnitude. This local design improvement would reduce the final W_f of 6,622 lb by approximately 45 lb.

As seen in Tables 9 and 10, the lower boundaries for T_{t4} , m_0 and C_{TO} , and the higher limit for the bypass ratio, were active. This indicates an even better design might have been selected if the boundaries for these variables had been expended appropriately, i.e., decreasing T_{t4} , m_0 , and C_{TO} lower limits and increasing the bypass ratio upper limit. It also suggests that there are significant differences between performance requirements for the SRI 3-leg and the SRI 19-leg missions, as confirmed by the results from obtained in Case 3 below.

Case 2: Short Range Interceptor On-design Mono-Objective Optimization

The mono-objective on-design optimization at $M_0 = 1.5$ and $h = 30,000$ ft for the SRI was successful and proceeded without interruption. The design variables and scaling parameters boundaries are listed in Table 11. As it was a mono-objective case, the value for the specific fuel consumption was not scaled. The results and performance for the optimum design are presented in Table 12. It took less than half an hour for the optimization codes to converge to a solution. This case was investigated to see how the

on-design solution compared with the full mission design obtained in Table 8. Feasible points were found within the first generation and were improved upon in the next 49 generations. This result was to be expected since on-design optimization was not heavily restricted due to the fact that there were no mission requirements to meet. The local optimizer successfully converged to a solution starting from the best solution provided by the global optimizer.

Table 11. Variables and Linear Scaling Boundaries for Case 2: SRI On-Design Optimization

<u>Design Variables Boundaries:</u>			
$20 \leq \Pi_c \leq 30$	$3 \leq \Pi_c \leq 4.5$	$2,900^\circ \text{ R} \leq T_{t4} \leq 3,200^\circ \text{ R}$	$3,200^\circ \text{ R} \leq T_{t7} \leq 3,600^\circ \text{ R}$
$0.2 \leq \alpha \leq 1$	$190 \text{ lb/s} \leq m_0 \leq 230 \text{ lb/s}$	$0.01 \leq C_{TO} \leq 0.02$	$0.2 \leq M_5 \leq 0.6$
<u>Linear Scaling Boundaries:</u>			
$1 \text{ (1/hr)} \leq S \leq 2.4 \text{ (1/hr)}$			

The global optimizer solution was close to the optimal solution since the local optimizer decreased the value of the fuel consumption by only 0.03% (from $S = 1.613$ to $S = 1.608$). Table 12 also clearly shows that the point provided by the global optimizer and the final design are very close. The Π_c and C_{TO} constraints were active, which suggested that these boundaries could have been expanded. The value of T_{t7} (3200° R) at the design flight conditions indicates that the afterburner was off and that there was enough dry thrust to meet the requirements at these flight conditions.

Table 12. Final Results for Case 2: SRI On-Design Optimization

<u>Best point from global optimizer at design point:</u>				
$\Pi_c = 23.1$	$\Pi_{c'} = 4.5$	$T_{t4} = 3,200^\circ \text{ R}$	$T_{t7} = 3,200^\circ \text{ R}$	$\alpha = 0.2$
$m_0 = 228.98 \text{ lb/s}$	$C_{TO} = 0.0141$	$M_5 = 0.26$	Function = 1.6913	
<u>Final design from local optimizer at design point:</u>				
$\Pi_c = 23.2$	$\Pi_{c'} = 4.5$	$T_{t4} = 3,200^\circ \text{ R}$	$T_{t7} = 3,200^\circ \text{ R}$	$\alpha = 0.2$
$m_0 = 220 \text{ lb/s}$	$C_{TO} = 0.01$	$M_5 = 0.51$	Function = 1.6908	
<u>Final performance (sea-level static):</u>				
$\Pi_c = 26.32$	$\Pi_{c'} = 4.97$	$T_{t4} = 2,935^\circ \text{ R}$	$T_{t7} = 3,600^\circ \text{ R}$	$\alpha = 0.185$
$m_0 = 279.1 \text{ lb/s}$	$C_{TO} = 0.0131$	$M_5 = 0.26$	$F_{SLmax} = 34,971 \text{ lb}$	
$S_{max} = 1.628 \text{ (1/hr)}$		$F/m_{0max} = 125.32 \text{ (lb/lbm/s)}$		$\Pi_{tL} = 0.4817$
$\Pi_{cH} = 5.297$		$\Pi_{tH} = 0.379$		

This point was different than the original design presented in Table 8. Only the value for T_{t4} and Π_c are similar. A mission analysis with the full SRI mission was performed with this final on-design solution. As expected, it failed the mission at several legs, the first one being the horizontal acceleration. This confirmed the limited usefulness of on-design optimization to find engine designs that can complete a given mission.

Case 3: Short Range Interceptor Mono-Objective Mission Optimization

The mono-objective 19-leg mission SRI case proceeded without interruption for 50 generations but failed to find a feasible point. No local optimization was performed

since the local optimizer could not operate from an unfeasible mission point. This case was investigated to compare the solution obtained from the automated optimization process with the one obtained from the trial-and-error method (see Table 8).

The engine and aircraft parameters for this case were set as in Case 1, the SRI test case. The aircraft cost and annulus area sub-objectives were deactivated. Moreover, the penalty for failure to achieve τ_{IL} convergence was increased to ensure that the optimizer would stay away from designs that failed this constraint. This protection was necessary since performance for these designs was unpredictable, it was hard for the global optimizer to correct the situation, and those designs took by far the longest time to solve. The penalty for the τ_{IL} constraint was applied in the same way as all other penalties, with the exceptions of the coefficient at the front of Equation (9) and the exponent of the same equation, which were set to 2.5 and 3 respectively.

The design variables and scaling parameters boundaries are listed in Table 13. After several unsuccessful runs, it was decided to bound the design variable with very narrow ranges, based on the solution presented in Table 8 in order to increase the probability of the global optimizer of finding a solution with a small initial population. This measure was necessary to keep the run time as low as possible. Since the problem was already heavily constrained by the difficult mission and the maximum Mach/maximum altitude leg, no constraint were imposed on minimum thrust or minimum weight (Equation (30)). The best unfeasible design obtained by the global optimizer is included in Table 14. It took three days for the optimization codes to complete the 50 generations.

This case was considered the ultimate test for the optimization scheme developed in this project. As realized previously by the author, it was a heavily constrained case with the $M = 2.5$ and $h = 50,000$ ft maximum Mach/maximum altitude leg being by far the most difficult constraint to meet. The 10,000 ft low altitude loiter leg was the other significant leg to meet. In fact, the search for a design that met these two legs would drive the whole global optimization process, as it did with the trial-and-error method. The feasible design space for the SRI full mission case was found to be so restricted that only a very narrow range of values for the design variables would meet all mission requirements. For example, the range of possible Π_c , as previously investigated, was less than ± 0.1 .

Table 13. Variables and Linear Scaling Boundaries for Case 3: SRI Mission Optimization

<u>Design Variables Boundaries, Original Case:</u>			
$25.6 \leq \Pi_c \leq 26.4$	$4.1 \leq \Pi_{c'} \leq 4.2$	$2,950^\circ \text{ R} \leq T_{t4} \leq 2,970^\circ \text{ R}$	$3,580^\circ \text{ R} \leq T_{t7} \leq 3,600^\circ \text{ R}$
$0.5 \leq \alpha \leq 0.65$	$259 \text{ lb/s} \leq m_0 \leq 263 \text{ lb/s}$	$0.014 \leq C_{TO} \leq 0.017$	$0.33 \leq M_5 \leq 0.37$
<u>Design Variables Boundaries, Modified Case ($M_{\max} = 2.3$):</u>			
$24 \leq \Pi_c \leq 28$	$3.5 \leq \Pi_{c'} \leq 4.5$	$2,900^\circ \text{ R} \leq T_{t4} \leq 3,100^\circ \text{ R}$	$3,550^\circ \text{ R} \leq T_{t7} \leq 3,600^\circ \text{ R}$
$0.4 \leq \alpha \leq 0.8$	$255 \text{ lb/s} \leq m_0 \leq 265 \text{ lb/s}$	$0.013 \leq C_{TO} \leq 0.02$	$0.3 \leq M_5 \leq 0.4$
$7,950 \text{ lb} \leq W_{fdes} \leq 8,500 \text{ lb}$			
<u>Linear Scaling Boundaries:</u>			
$6,000 \text{ lb} \leq W_f \leq 18,000 \text{ lb}$	$\$10,000,000 \leq \text{Cost} \leq \$30,000,000$	$0.69 \text{ ft}^2 \leq \text{Area} \leq 11.02 \text{ ft}^2$	

The optimization process failed to find a feasible solution because no feasible solution exists for this case when using the Matlab engine analysis codes developed for

this project, for reasons covered in the next paragraph. This conclusion was reached because the final design presented in Table 8 failed the mission when the reference point was adjusted to sea-level static. The design from Table 8 successfully passed the mission when its reference flight conditions were set to their original values of $M = 1.6$, $h = 35,000$ ft. However, when the solution at sea-level static for this engine (obtained with off-design analysis) was tried as reference point, it failed the mission. The loiter and maximum Mach/maximum altitude legs were not met because τ_{iL} did not converge.

Table 14. Final Results for Case 3: SRI Mission Optimization

<u>Best unfeasible point from global optimizer, original case:</u>				
$\Pi_c = 25.92$	$\Pi_c = 4.12$	$T_{t4} = 2,959.2^\circ \text{ R}$	$T_{t7} = 3,582.8^\circ \text{ R}$	$\alpha = 0.51$
$m_0 = 259.6 \text{ lb/s}$	$C_{TO} = 0.0167$	$M_5 = 0.338$	Function = -2.7362	
<u>Best point from global optimizer, modified case ($M_{\max} = 2.3$):</u>				
$\Pi_c = 26.13$	$\Pi_c = 4.02$	$T_{t4} = 3,039^\circ \text{ R}$	$T_{t7} = 3,558.7^\circ \text{ R}$	$\alpha = 0.652$
$m_0 = 263.71 \text{ lb/s}$	$C_{TO} = 0.013$	$M_5 = 0.359$	Function = 0.8132	
<u>Final design from local optimizer, modified case ($M_{\max} = 2.3$):</u>				
$\Pi_c = 26.23$	$\Pi_c = 4.01$	$T_{t4} = 3,003.6^\circ \text{ R}$	$T_{t7} = 3,555.1^\circ \text{ R}$	$\alpha = 0.652$
$m_0 = 263.97 \text{ lb/s}$	$C_{TO} = 0.013$	$M_5 = 0.359$	Function = 0.8152	
<u>Final performance, modified case ($M_{\max} = 2.3$, sea-level static):</u>				
$M_{0\max} = 2.3$	$h_{\max} = 50,000 \text{ ft}$			
$F_{SL\max} = 30,737 \text{ lb}$	$F_{SLmil} = 19,306 \text{ lb}$	$S_{\max} = 1.733 \text{ (1/hr)}$	$S_{mil} = 0.849$	
$F/m_{0\max} = 116.44 \text{ (lbf/lbm/s)}$		$F/m_{0mil} = 73.14 \text{ (lbf/lbm/s)}$		
$\Pi_{iL} = 0.419$	$\Pi_{cH} = 6.533$	$\Pi_{iH} = 0.352$	$P_{TO} = 450,040 \text{ W}$	
Cost = \$14,472,000	Area = 7.13 ft ²	$W_{TO} = 25,171 \text{ lb}$	$W_f = 8,306 \text{ lb}$	

As mentioned above, small variations in the engine analysis calculations performed by the codes developed by the author and the calculations obtained from the ONX, OFFX, and MISS programs, are believed to account for the failure of the original design when the reference conditions are taken as sea-level static. Examples of potential variations are the different altitude models used and the more relaxed convergence criterion in 'offxmiss.m'. These differences in calculations become a factor because the ranges for the design variables at $M = 2.5$ and $h = 50,000$ ft are very narrow. Moreover, as mentioned before, the design space for this case is very restricted, even when using Mattingly's codes. This means small variations in calculations at sea-level may have a large enough impact at the maximum Mach/maximum altitude leg flight conditions to turn a feasible design into an infeasible one.

This hypothesis was confirmed with the performance of an off-design analysis at the original SRI reference point of 35,000 ft and $M = 1.6$ with the sea-level static design from Table 8 as reference point. The results of this analysis are presented in Table 15, along with the original design values from Table 8. It can be seen that, as expected, the design variables values obtained with the codes developed in this study are practically the same as those provided in Table 8. However, the small differences between the original values and the calculated values were enough to make the design obtained from the off-design analysis fail both the loiter and maximum Mach/maximum altitude leg.

Table 15. SRI Design Variables Values at $M = 1.6$ and $h = 35,000$ ft

<u>Trial-and-Error Process Original design values:</u>			
$\Pi_c = 22.5$	$\Pi_c = 3.7$	$T_{t4} = 3,200^\circ \text{ R}$	$T_{t7} = 3600^\circ \text{ R}$
$\alpha = 0.6$	$m_0 = 204 \text{ lb/s}$	$C_{TO} = 0.016$	$M_5 = 0.35$
<u>Off-design analysis values:</u>			
$\Pi_c = 22.5055$	$\Pi_c = 3.7051$	$T_{t4} = 3,200^\circ \text{ R}$	$T_{t7} = 3,600^\circ \text{ R}$
$\alpha = 0.5946$	$m_0 = 203.7545 \text{ lb/s}$	$C_{TO} = 0.016$	$M_5 = 0.3508$

The gross weight, W_{TO} , was not updated to a final value, since no feasible point were found to start the convergence process. The initial W_{TO} for this case was set to 24,800 lb, the final value previously obtained by the author.

Although the final design is infeasible, it can be seen from Table 14 that the final design is close to the original solution. To determine how close the final solution was to the feasible region, the maximum Mach/maximum altitude leg requirement was relaxed gradually until the final design met the modified mission (using the 'offx.m' stand-alone off-design analysis program). This leg was selected since it was the only constraint that the design did not meet. The maximum Mach/maximum altitude leg requirements was eased up by decreasing the maximum Mach number. The highest Mach number to allow the final design from Table 14 to be feasible was $M = 2.34$ at $h = 50,000$ ft.

In light of the fact that the final design obtained with the trial-and-error process failed the mission when sea-level static was the reference point, the global optimizer did improve the original design by bringing the design closer to the feasible region. While the

original design failed both the loiter and the maximum Mach/maximum altitude leg due to τ_{IL} convergence, the solution provided by the optimizer only failed the maximum Mach/maximum altitude leg because the low pressure turbine might have been unchoked. Moreover, even though the optimization code did not find a solution, it provided useful design information by clearly indicating which part of the mission could not be met and why.

Modified SRI Full Mission Optimization. Based on this above information, Case 3 was retried with a maximum Mach requirement reduced to $M_{\max} = 2.3$. This case was investigated to see how the optimizer would behave with a feasible but very restricted case. The number of generations was set to 25 and the design variable bounds were expanded, as indicated in Table 13, to allow the search of a larger design space.

This case failed when a W_{TO} convergence model based on Equations (10), (11), and (12) was used because the gross weight diverged upward until no feasible designs could be found. This occurrence demonstrated how sensitive this weight convergence method is to the initial W_{TO} value.

On the other hand, this case was resolved successfully with the use of the W_{TO} determination method based on Equations (14) and (15). A feasible design was found at generation 2. Although the local optimizer was able to improve the best solution from the genetic algorithm, it oscillated between the feasible and unfeasible, which might suggest it was close to failure. As seen from Table 14, the improvement on the objective function value was 0.25%, from 0.8132 to 0.8152. The behavior of the local optimizer is

due to the fact that the final solution was very close to the unfeasible mission region as it was noted that only a small range of designs would prove feasible. This situation would force the optimizer to step into the infeasible mission region as it tries to update the design.

The final design and its associated performance is presented in Table 14. It should be noted that the design and performance of the final engine obtained from the optimizer are close to the original results presented in Table 8.

As seen in Tables 13 and 14, none of the design variable side constraints were active, with the exception of C_{TO} which was at its lower limit in both the global and local solutions. This indicated the variables were well bounded for this case, although the lower limit for C_{TO} could have been lowered further.

Case 4: Global Strike Aircraft Mono-Objective Mission Optimization

The mono-objective 19-leg mission GSA case was successful. This case was investigated to provide a comparison baseline for Case 5 (multiobjective GSA), in order to evaluate the impact of the aircraft cost and engine annulus area sub-objectives. This case was also useful to determine the validity of the W_{TO} convergence scheme for the GSA described in Chapter 2. The design variables and scaling parameters boundaries were set as listed in Table 16, based on data from Reference 16. Since no hard data were provided in Reference 4, it was decided to set the high altitude turns at 1.5g and the loiter leg at 30,000 ft. These values were selected in order to avoid overconstraining the mission. With data from Table 7.2 of Reference 4, the minimum weight without fuel, W_{min} , was set

to 92,155 lb. The results and performance for the optimum design are presented in Table 17. As it was noticed in test runs that feasible points were found early and that the gross weight seemed to converge quickly, the global optimizer was set to 25 generations instead of 50. However, as discussed below, the number of generations should have been higher. It took two days for the optimization codes to converge to a solution. Feasible points were found within the second generation.

Table 16. Variables and Linear Scaling Boundaries for Case 4: GSA Mono-Objective Mission Optimization.

<u>Design Variables Boundaries:</u>			
$80 \leq \Pi_c \leq 100$	$3 \leq \Pi_c \leq 8.5$	$4,000^\circ \text{ R} \leq T_{t4} \leq 4,460^\circ \text{ R}$	$0^\circ \text{ R} \leq T_{t7} \leq 0^\circ \text{ R}$
$0.5 \leq \alpha \leq 2$	$400 \text{ lb/s} \leq m_0 \leq 470 \text{ lb/s}$	$0.005 \leq C_{TO} \leq 0.02$	$0.2 \leq M_5 \leq 0.6$
<u>Linear Scaling Boundaries:</u>			
$110,000 \text{ lb} \leq W_f \leq 320,000 \text{ lb}$			

As indicated in Table 17, the global optimizer provided only two points to the local optimizer, and not the three best designs, as would be expected. It should also be noted that the second best point is nearly identical to the first. The reasons for this situation are covered in Case 1. The best design did not improve much in the last 3 generations, even as W_{TO} was updated, indicating that the global optimizer was near the best design and that it was appropriate at this point to switch to the local optimizer.

Table 17. Final Results for Case 4: GSA Mono-Objective Mission Optimization

<u>Best point from global optimizer:</u>					
$\Pi_c = 80.54$	$\Pi_c = 8.13$	$T_{t4} = 4,373.9^\circ \text{ R}$	$T_{t7} = 0^\circ \text{ R}$	$\alpha = 0.808$	
$m_0 = 416.34 \text{ lb/s}$	$C_{TO} = 0.009$	$M_5 = 0.323$	Function = 0.99978		
<u>Second best point from global optimizer:</u>					
$\Pi_c = 80.54$	$\Pi_c = 8.13$	$T_{t4} = 4,374.3^\circ \text{ R}$	$T_{t7} = 0^\circ \text{ R}$	$\alpha = 0.808$	
$m_0 = 416.34 \text{ lb/s}$	$C_{TO} = 0.009$	$M_5 = 0.323$	Function = 0.99953		
<u>Third best point from global optimizer:</u>					
$\Pi_c = 0$	$\Pi_c = 0$	$T_{t4} = 0$	$T_{t7} = 0$	$\alpha = 0$	
$m_0 = 0$	$C_{TO} = 0$	$M_5 = 0$	Function = N/A		
<u>Final design from local optimizer:</u>					
$\Pi_c = 80.54$	$\Pi_c = 8.13$	$T_{t4} = 4,374.3^\circ \text{ R}$	$T_{t7} = 0^\circ \text{ R}$	$\alpha = 0.808$	
$m_0 = 416.34 \text{ lb/s}$	$C_{TO} = 0.009$	$M_5 = 0.323$	Function = 0.99953		
<u>Final performance (sea-level static):</u>					
$F_{SLmil} = 42,509 \text{ lb}$	$S_{mil} = 0.8273 \text{ (1/hr)}$	$F/m_{omil} = 102.1 \text{ (lbf/lbm/s)}$	$\Pi_{tL} = 0.323$		
$\Pi_{cH} = 9.91$	$\Pi_{tH} = 0.32$	$P_{TO} = 492,621 \text{ W}$			
Cost = \$97,808,200	Area = 10.41 ft ²	$W_{TO} = 202,964 \text{ lb}$	$W_f =$		
110,179 lb					

The local optimizer did not improve the best solution from the genetic algorithm and eventually diverged into the infeasible mission region. The reason for this behavior of the local optimizer was covered in Case 3. The improvement on the objective function value was negligible, of the order of 1×10^{-10} and the final design was the same as the one provided by the genetic algorithm. This indicates that the global optimizer had converged on the final solution.

The gross weight, W_{TO} , did not converge to a final value, although it was close to doing so. The initial W_{TO} for this case was set to 259,500 lb. The final value for W_{TO} was

202,964 lb and it was practically at the lower W_{TO} limit, as determined by Equations (29), (30), and the minimum W_f limit of 110,000 lb. It should be noted that W_{TO} was updated every generation until the end of the run at generation 25. This demonstrated that the genetic algorithm was able to significantly improve the best design throughout the run in order to lower the fuel weight and hence, the gross weight. It also demonstrated the advantages of a very high technology engine.

It can be seen that the results presented in Table 17 are of the same order as what was estimated in Reference 4, although one must keep in mind that high temperature effects were neglected, as stated in Chapter 3. It was also verified that all the constraints were met.

The altitude properties algorithms developed as part of the optimization codes, which were designed for altitude of up to 65,000 ft (the limit of the isothermal layer), were used for this case, even though the GSA mission included legs at altitude above 70,000 ft. It was decided to do so since the standard atmosphere table in Appendix B of Reference 10 shows that the isothermal layer could extend up to 80,000 ft, instead of the 65,000 ft stated in Reference 2.

The aircraft cost of \$97.8 million is low compared to the cost of \$150 million estimated in Reference 4. This can be attributed to the crudeness of the cost model, as explained in Chapter 2. However, as the objective function is only required to minimize cost in order to find the best engine design, an accurate value was not necessary.

As seen in Tables 16 and 17, none of the design variable side constraints were active, although the bypass ratio (α) and the compressor pressure ratio (Π_c) were close to

their lower and upper limits respectively. This indicated the variables were well bounded for this case. The fact that the final fuel weight is so close to its lower limit suggest that a better design might have been discovered if this limit had been lowered even further.

Case 5: Global Strike Aircraft Multiobjective Mission Optimization

The multiobjective 19-leg mission GSA case was successful and proved the feasibility of the full multiobjective optimization process for a complex case. This case was investigated to determine the impact of cost and area on the final engine design, as compared with Case 4 (mono-objective GSA). This case was set up in exactly the same way as Case 4, with the exception that the aircraft cost and annulus area sub-objectives, which were active. Thirty generations were used for the global optimization, instead of the 25 used for the GSA mono-objective case, in order to ensure W_{TO} had converged. The design variables and scaling parameters boundaries are listed in Table 18 and they are again based on data from Reference 16. The results and performance for the optimum design are presented in Table 19. It took two days for the optimization codes to converge to a solution. Feasible points were found within the third generation.

Table 18. Variables and Linear Scaling Boundaries for Case 5: GSA Multiobjective Mission Optimization

<u>Design Variables Boundaries:</u>			
$80 \leq \Pi_c \leq 100$	$3 \leq \Pi_{c'} \leq 8.5$	$4,000^\circ \text{ R} \leq T_{t4} \leq 4,460^\circ \text{ R}$	$0^\circ \text{ R} \leq T_{t7} \leq 0^\circ \text{ R}$
$0.5 \leq \alpha \leq 3$	$400 \text{ lb/s} \leq m_0 \leq 470 \text{ lb/s}$	$0.005 \leq C_{TO} \leq 0.02$	$0.2 \leq M_5 \leq 0.6$
<u>Linear Scaling Boundaries:</u>			
$175,000 \text{ lb} \leq W_f \leq 320,000 \text{ lb}$	$\$65,000,000 \leq \text{Cost} \leq \$150,000,000$		
$5 \text{ ft}^2 \leq \text{Area} \leq 150 \text{ ft}^2$			

Table 19. Final Results for Case 5: GSA Multiobjective Mission Optimization

<u>Best point from global optimizer:</u>				
$\Pi_c = 81.16$	$\Pi_{c'} = 8.42$	$T_{t4} = 4,373.3^\circ \text{ R}$	$T_{t7} = 0^\circ \text{ R}$	$\alpha = 0.795$
$m_0 = 409.68 \text{ lb/s}$	$C_{TO} = 0.0105$	$M_5 = 0.2$	Function = 1.83109	
<u>Second best point from global optimizer:</u>				
$\Pi_c = 0$	$\Pi_{c'} = 0$	$T_{t4} = 0$	$T_{t7} = 0$	$\alpha = 0$
$m_0 = 0$	$C_{TO} = 0$	$M_5 = 0$	Function = N/A	
<u>Third best point from global optimizer:</u>				
$\Pi_c = 0$	$\Pi_{c'} = 0$	$T_{t4} = 0$	$T_{t7} = 0$	$\alpha = 0$
$m_0 = 0$	$C_{TO} = 0$	$M_5 = 0$	Function = N/A	
<u>Final design from local optimizer:</u>				
$\Pi_c = 81.16$	$\Pi_{c'} = 8.42$	$T_{t4} = 4,373.3^\circ \text{ R}$	$T_{t7} = 0^\circ \text{ R}$	$\alpha = 0.795$
$m_0 = 409.68 \text{ lb/s}$	$C_{TO} = 0.0105$	$M_5 = 0.2$	Function = 1.83109	
<u>Final performance (sea-level static):</u>				
$F_{SLmil} = 41,959 \text{ lb}$	$S_{mil} = 0.8289 \text{ (1/hr)}$	$F/m_{0mil} = 102.42 \text{ (lbf/lbm/s)}$		
$\Pi_{tL} = 0.316$	$\Pi_{cH} = 9.64$	$\Pi_{tH} = 0.3221$	$P_{TO} = 565,531 \text{ W}$	
Cost = \$97,782,840	Area = 10.24 ft ²	$W_{TO} = 212,218 \text{ lb}$	$W_f =$	
110,413 lb				

As indicated in Table 19, the global optimizer provided only one point to the local optimizer. The reasons for this situation are as discussed in previous cases. The best design did not improve beyond generation 26, even as W_{TO} kept decreasing, as explained below.

The local optimizer behaved as in Case 4 and failed to improve the best solution from the genetic algorithm and eventually diverged into the infeasible mission region. The improvement on the objective function value was negligible, of the order of 1×10^{-5} and the final design was the same as the one provided by the genetic algorithm. The reasons for this behavior are explained in the section that covers Case 3 above.

The final value of W_{TO} for the best design was 212,964 lb and the fuel weight was close to its lower limit of 110,000 lb. It should be noted that this value of W_{TO} was determined at generation 26 and was not the lowest value for the gross weight. The gross weight was updated and decreased, to a value of 203,739 lb, until generation 28, after which the optimizer failed to return to the feasible region in the last two generations. This result suggests that although genetic algorithm was able to significantly improve the best engine design, the area and cost sub-objectives limited the potential decrease in gross weight, since W_{TO} for the multiobjective optimization is 9,254 lb higher than for the mono-objective case. The impact of these factors is described in more detail below.

With the addition of the aircraft cost sub-objective to the objective function, the aircraft cost was reduced from \$97,808,200 to \$97,782,840 between Case 4 and Case 5, a 0.025% decrease.

With the application of the area sub-objective, the annulus area was reduced from 10.41 ft² (see Table 16) to 10.24 ft², a 1.63% improvement. Since the relative improvement in area, although small, is two orders of magnitude larger than the improvement in cost, it is considered that the engine annulus area sub-objective was the principal cause for the difference between Case 4 and this case. The impact of the area requirement is illustrated by comparing the values for W_f and W_{TO} obtained for each case in Table 17 and 19. W_f increased by 0.21%, from 110,179 lb to 110,413 lb. W_{TO} increased from 202,965 lb to 212,218 lb, a 4.56% change. The discrepancy between the relative changes in W_f and W_{TO} are due to the minimum weight constraint imposed by Equation (31). At $W_{TO} = 202,965$ lb, the design was very close to fail this constraint and thus was limited on the potential improvement of W_f . On the other hand, for the case where W_{TO} is 212,218 lb, this constraint is not as critical as the optimizer had approximately 10,000 more pounds of gross weight to play with before it hit the minimum weight constraint. It can also be observed from Table 16 and 18 that the final designs for Case 4 and Case 5 are similar, although they shows significant difference in their values of Π_c and M_5 (6.89 vs. 5.19 and 0.323 vs. 0.2 respectively). This led to a 1.29% reduction in thrust, from 42,509 lb to 41,959 lb. This reduction in thrust for an increase in fuel usage indicates that the multi-objective design is not as efficient as the mono-objective one. However, it is not a critical factor as it was observed that the thrust generated by these designs was far above what was required to accomplish the mission.

It can be concluded from the small difference in performance between the mono-objective and multiobjective cases investigated that the mission fuel weight has by far the most impact on the engine final design.

As seen in Table 19, only the M_5 side constraint was active. This indicated the variables were well bounded for this case.

Case 6: Global Strike Aircraft Multiobjective On-Design Optimization

The mono-objective on-design optimization at $M_0 = 1.5$ and $h = 60,000$ ft for the GSA was successful and proceeded without interruption. The design variables and scaling parameter boundaries are listed in Table 20. Several trial runs, including stand-alone off-design analysis and mission analysis, were required to establish these boundaries at the design flight conditions. The results and performance for the optimum design are presented in Table 21. It took less than half an hour for the optimization codes to converge to a solution. This case was investigated to see how the on-design solution compared with the full mission design obtained in case 5 (see Table 19). Feasible points were found within three generations and were improved upon in the next 47 generations. The local optimizer successfully converged to a solution starting from the best solution provided by the global optimizer.

Table 20. Variables and Linear Scaling Boundaries for Case 6: GSA On-Design Optimization

<u>Design Variables Boundaries:</u>			
$60 \leq \Pi_c \leq 90$	$2 \leq \Pi_{c'} \leq 4$	$4,000^\circ \text{ R} \leq T_{t4} \leq 4,460^\circ \text{ R}$	$0^\circ \text{ R} \leq T_{t7} \leq 0^\circ \text{ R}$
$1 \leq \alpha \leq 3$	$100 \text{ lb/s} \leq m_0 \leq 300 \text{ lb/s}$	$0.01 \leq C_{TO} \leq 0.03$	$0.2 \leq M_5 \leq 0.6$
<u>Linear Scaling Boundaries:</u>			
$120,000 \text{ lb} \leq W_f \leq 320,000 \text{ lb}$	$\$75,000,000 \leq \text{Cost} \leq \$150,000,000$		
$6.2 \text{ ft}^2 \leq \text{Area} \leq 44.11 \text{ ft}^2$			

The global optimizer best solution underwent a major improvement when fed to the local optimizer. The value of the objective function increased by 9.5%, from 1.6215 to 1.7726. Table 21 also clearly shows that the point provided by the global optimizer and the final design are radically different. The α , C_{TO} and M_5 constraints were active, which suggested that these boundaries could have been expanded.

A mission analysis with the full SRI mission was not performed with this final on-design solution since it failed the off-design analysis at sea-level static, the reference conditions used for comparison with Case 5. This is another clear example where on-design optimization at a given flight condition is of little help in finding an engine design that will complete a given mission. The sea-level values are presented in Table 21, but the data is not to be considered valid since α' did not converge. It can be observed that most of the sea-level data is unrealistic, with, for example, very low values of Π_c , $\Pi_{c'}$ and F/m_0 .

Table 21. Final Results for Case 6: GSA On-design Optimization

<u>Best point from global optimizer at design point:</u>					
$\Pi_c = 64.2$	$\Pi_{c'} = 2.18$	$T_{t4} = 4,293^\circ \text{ R}$	$T_{t7} = 0^\circ \text{ R}$	$\alpha = 1.27$	
$m_0 = 103.54 \text{ lb/s}$	$C_{TO} = 0.0293$	$M_5 = 0.439$	Function = 1.6215		
<u>Final design from local optimizer at design point:</u>					
$\Pi_c = 80.3$	$\Pi_{c'} = 3.11$	$T_{t4} = 4,380^\circ \text{ R}$	$T_{t7} = 0^\circ \text{ R}$	$\alpha = 3$	
$m_0 = 101.17 \text{ lb/s}$	$C_{TO} = 0.03$	$M_5 = 0.2$	Function = 1.7726		
<u>Final performance (sea-level static):</u>					
$\Pi_c = 37.2$	$\Pi_{c'} = 1.72$	$T_{t4} = 2,970^\circ \text{ R}$	$T_{t7} = 0^\circ \text{ R}$	$\alpha = 7.08$	
$m_0 = 536.78 \text{ lb/s}$	$C_{TO} = 0.03$	$M_5 = 0.24$	$F_{SLmil} = 14,997 \text{ lb}$		
$S_{mil} = 0.436 \text{ (1/hr)}$		$F/m_{0mil} = 27.94 \text{ (lbf/lbm/s)}$		$\Pi_{tL} = 0.175$	
$\Pi_{cH} = 21.61$		$\Pi_{tH} = 0.205$			
<u>Note:</u> Results at sea-level not valid since design failed at these conditions (see text)					

V. Conclusions and Recommendations

Introduction

The purpose of this thesis was to develop tools that would automate to a large extent aircraft engine conceptual design optimization. This goal was accomplished with the integration of engine cycle and mission analysis with optimization technique such as sequential quadratic programming and genetic algorithms. Various sample cases such as a short range interceptor and a global strike aircraft were investigated as proof of concept. Overall, the objectives stated in Chapter 3 were achieved, for both on-design and mission optimization, with optimal solutions obtained in all cases investigated. The general conclusions derived from the thesis results and the recommendations for future work on this project are presented below.

Conclusions

With the integration of the engine analysis and optimization code under the same programming architecture, conceptual engine design for on-design and mission optimization was successfully automated. The optimization process ensured that all mission requirements and design limitations were met and that the final engine designs were the smallest, the most fuel efficient, and with the least impact on total aircraft cost.

Since all codes are written in Matlab, which uses a simple language and file manipulation system, it is easy to modify, improve, and add to the existing codes. This modularity also makes it relatively simple to expand the engine analysis and optimization

tools to include more detail models, such as more advanced installation losses and cost models.

Although to set up a case only requires the manipulation of a few files, it is a process prone to mistakes; mistakes that could modify the original codes. Moreover, a good working knowledge of Matlab is required to manipulate the codes with confidence. For these reasons, the development of a fool-proof user interface is a must.

The combination of a genetic algorithm (GA) with a local optimizer proved effective, with the GA exploring a given design space to reach an often limited feasible region and the gradient based optimizer quickly improving, whenever possible, on the best solution provided by the GA. Improvement of the objective function value of up to 10% was achieved by the local optimizer. The use of the three best points from the GA did not prove useful, since the points were similar and would converge to the same solution by the end of the local optimization. This dual optimization technique is made even more important due to the fact that the design space appears to be a relatively flat surface pockmarked with small minimas and maximas. Constraints were successfully applied to the GA with the use of penalty functions. The codes developed for this thesis were very robust, with engine analysis codes that consistently provided usable data to the optimizers, even for highly infeasible designs.

On-design optimization based on one flight condition worked very well but, as it was of limited use since a design optimized for a given flight condition would usually fail other conditions. It was much less constrained than mission optimization, and feasible points would be found within three generations. The process would typically take less

than half an hour to be completed. On-design optimization proved useful to set up variable boundaries for mission optimization since on-design optimization is a fast process.

The optimization processes developed as part of this thesis were subject to certain limitations. For one, mission optimization for very constrained mission requirements, such as a low altitude loiter with a high maximum Mach/maximum altitude leg might not produce a feasible solution, as in the original SRI full mission case. Moreover, the local optimizer could not converge to a solution if started from an infeasible, or near infeasible, mission point. This problem could be alleviated with larger populations and more generations.

It is of critical importance to properly bound the design variables and the scaling limits to ensure a faster convergence to a feasible design. This is a process based on experience and trials runs are often required to determine suitable boundaries.

Another limitation is the possible requirement to setup a gross weight (W_{TO}) convergence scheme for each different aircraft under investigation. If an improper W_{TO} model is used, the aircraft gross weight may diverge away from the optimal solution if W_{TO} is not properly matched with W_F . On the other hand, for the cases investigated, the W_{TO} convergence schemes proved successful, although longer global optimization run would be necessary to ensure W_{TO} has fully converged. The W_{TO} determination model based on Equations (14) and (15), which assign an individual gross weight to each design, was considered superior to the model based on Equations (10) to (13) since it required little aircraft specific data. This model was also impervious to gross weight divergence and would allow considerably shorter runs.

A drawback of the mission optimization process is that it is slow, with full-mission cases requiring between two and three days to converge to a solution, although no user intervention was necessary during a run. also, it must should be pointed out that the process involves some major simplifications such as constant installation losses and a crude aircraft cost model. The cost model used produced low aircraft cost, especially in the case of the global strike aircraft. On the other hand, since the purpose of the cost model was to minimize engine impact on aircraft cost and not to evaluate the cost accurately, the model proved adequate.

Variations of the sub-objective weight factors, K_k , and the individual impacts of engine annulus area and aircraft cost were not investigated, although it was observed that fuel weight had, by far, the most impact on the final design. Engine annulus area was a distant second and the aircraft cost sub-objective had virtually no impact on the final design. This work is left to future users of the optimization codes.

Despite its limitations and simplifications, it is believed that the tools developed as part of this thesis are useful as they eliminate a lot of the manual groundwork and guesswork required at the early stage of conceptual engine design. The concepts brought forward during this project proved viable, although it is understood that the tools created to apply them requires further development. The solution provided at the end of the optimization process should provide a good first engine design which can then be improved upon.

Recommendations

It is believed that further work is required to allow the codes and processes developed as part of this thesis to reach their full potential as a powerful and versatile engine optimization tool.

First and foremost, a user-friendly interface must be created to efficiently and safely enter all the data, parameters, and models (such as cost, W_{TO} convergence, area, etc.) required to set up an optimization case.

The engine analysis codes, especially 'offxmiss.m', should be reviewed and modified as required to increase their speed in order to shorten the duration of the optimization runs. Alternative engine and mission analysis codes could be also integrated to the existing optimization codes in order to get a faster or more accurate optimization process. Such codes could be in Fortran or C since Matlab can readily integrate and process programs written in these languages. As a short term solution to the speed limitations, cases should be run on dedicated and fast computers. The codes m-files names should also be modified to allow them to work under a DOS environment.

Advanced models should be created or integrated for engine annulus area, and especially aircraft cost and installation losses. Development and inclusions of these models would bring this project closer to a comprehensive engine design optimizer.

The impacts of engine annulus area, aircraft cost, and objective weight factors, K_k , on the final design should be investigated further with additional runs set up with several combinations of K_1 , K_2 , and K_3 .

Appendix A: Instructions on How to Obtain Optimization and Engine Analysis
Computers Codes

In order to obtain copies of some of, or all, the Matlab computer codes developed and used as part of this study, contact:

Department of Aeronautics and Astronautics
School of Engineering
Air Force Institute of Technology
Wright-Patterson Air Force Base, Ohio
45433-6583

Appendix B: Drag Profiles

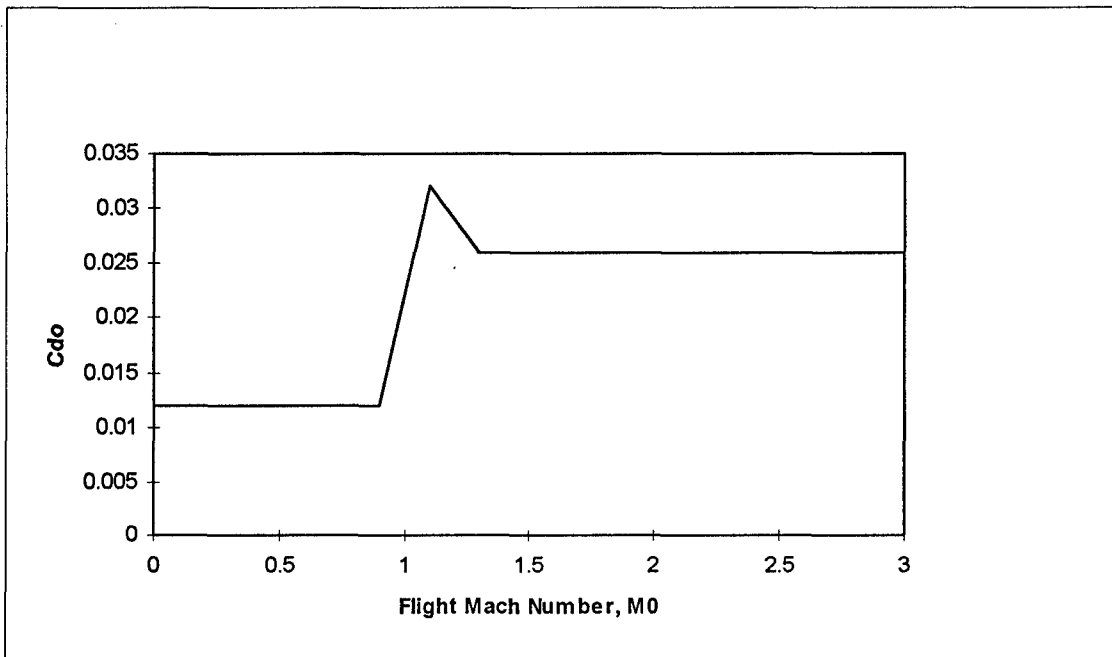


Figure 14. C_{D0} vs. Flight Mach Number, M_0 , for the Short Range Interceptor

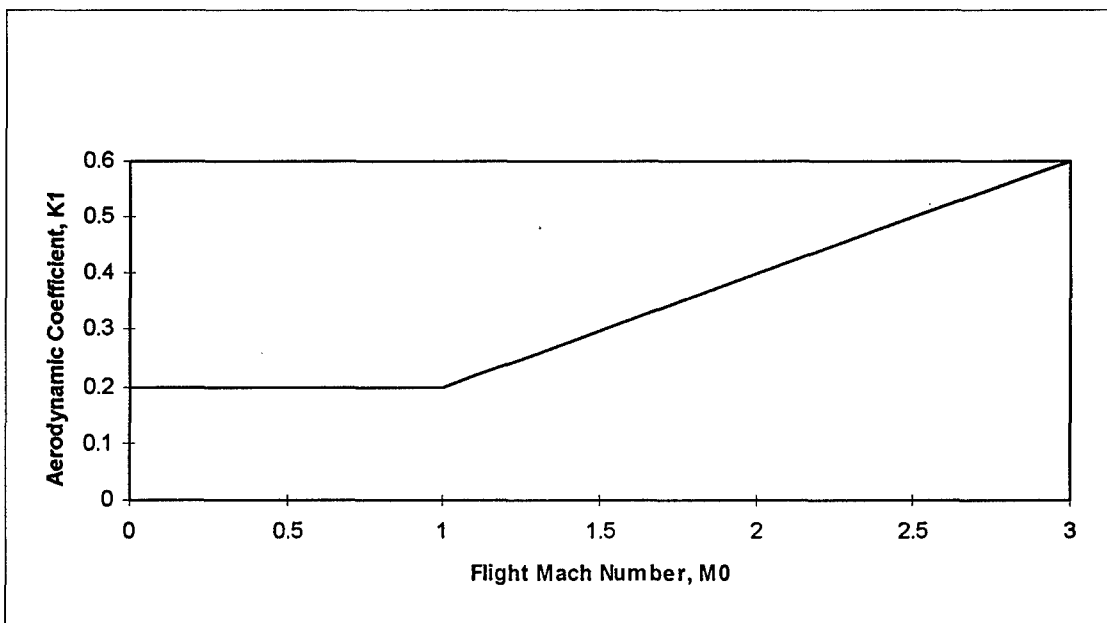


Figure 15. K_1 vs. Flight Mach Number M_0 , for the Short Range Interceptor

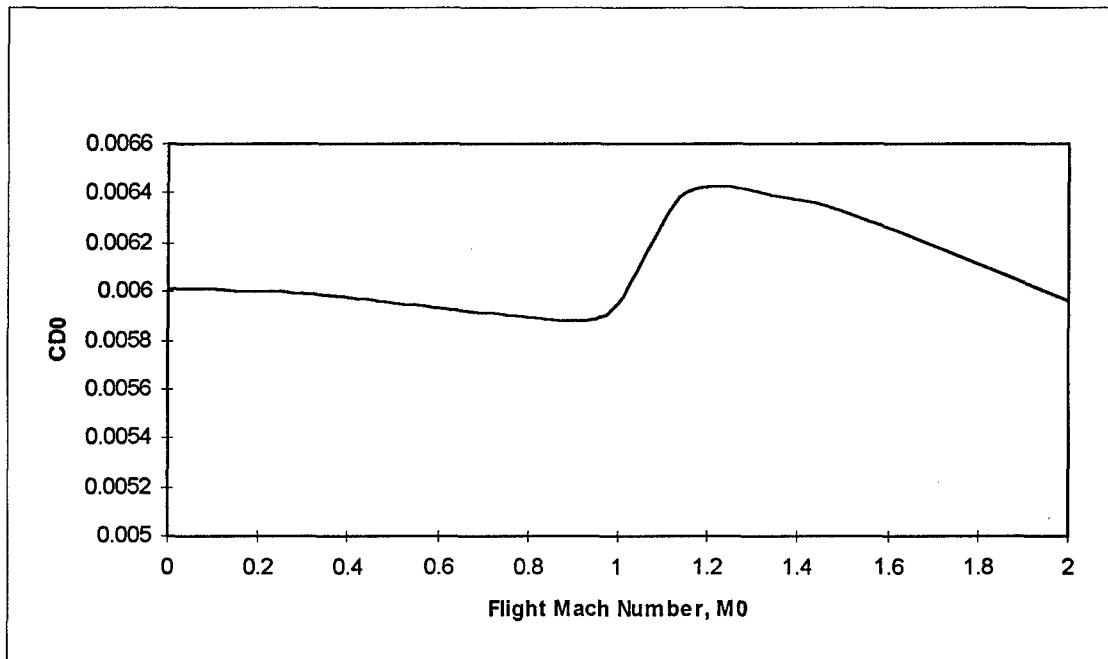


Figure 16. C_{D0} vs. Flight Mach Number, M_0 for the Global Strike Aircraft

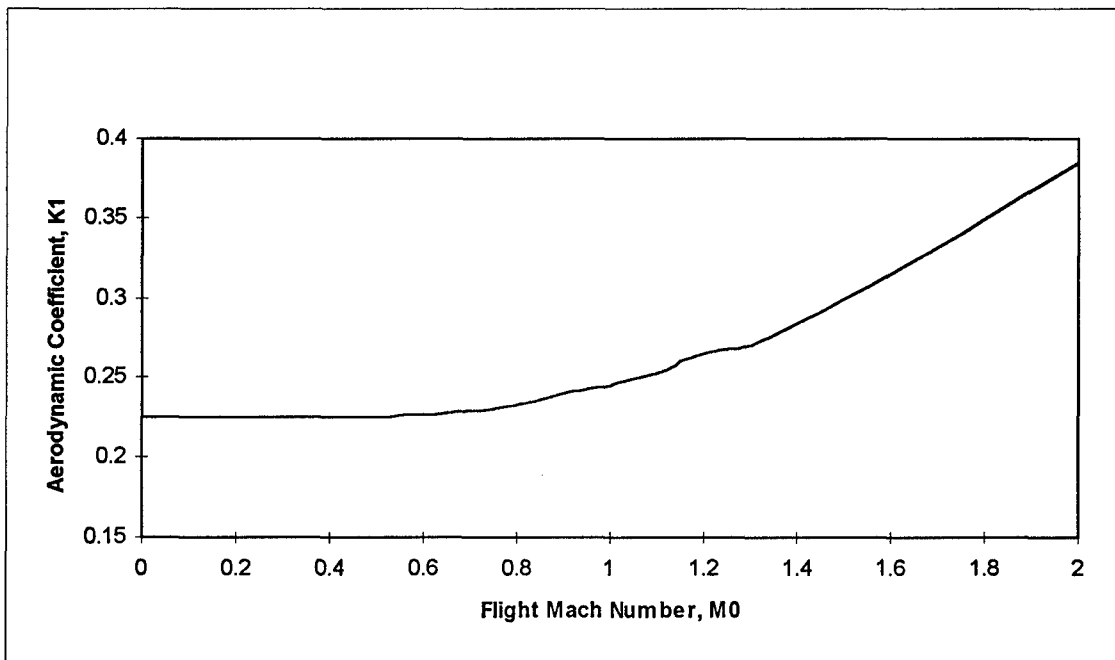


Figure 17. K_1 vs. Flight Mach Number, M_0 , for the Global Strike Aircraft

Appendix C: Engine Optimization Computer Codes Operating Instructions

The purpose of this document is to provide step-by-step instructions on how to use the provided engine analysis and optimization codes. Procedures are included to solve the following classes of problems:

1. On-design engine optimization at given flight conditions.
2. Engine optimization with mission.
3. Stand-alone on-design engine cycle analysis.
4. Stand-alone off-design engine cycle analysis.
5. Stand-alone mission analysis.

No built-in interface is available to enter variables and parameters. However, the process is simple in that it only involves accessing a few Matlab m-files to enter the data required to set up a case.

General Comments

All the required codes are included with the disk that comes with these instructions. The files have to be downloaded in the same directory. The codes include the optimizers and the engine analysis codes. The codes have been developed to work with the Unix operating system. To work under DOS, some files names would have to be modified to ensure that they are less than eight characters long.

It is assumed that the user has a basic knowledge of Matlab and on how to create Matlab m-files. The user should also be familiar with text editors, such as NEDIT, Vi or Windows' NOTEPAD. The instructions are set for Windows-type environment.

As a reminder, if one desires to disable a function or calculation (such as a constraint), all that is required is to open the appropriate file and type '%' at the beginning of the desired code line. To see the value of a variable that does not normally appear on the screen during a run, one must open the file, go to the required line and remove the semi-colon at the end of the line.

Annex A includes the description of all the variables of interest to the user. Annex B contains a list of all the m-files used for optimization and engine cycle analysis.

It should be noted that any variable with the suffix '_noAB' implies the value of the variable when the afterburner is off. The suffix 'of' is applied to all off-design variables and the subscript 'SL' refers to sea-level values.

Updating codes with Fortran or C routines. Matlab has the capability to integrate external Fortran or C codes and subroutines. This is a useful feature that allows a user to substitute any of the m-file used in the optimization process with his/her own. For example, a faster and more accurate existing Fortran off-design analysis code might be used instead of 'offxmiss.m'.

There are two relatively simple way to integrate external codes to a Matlab program. The first, and easiest to use, is the shell escape function. With this method the user write a simple m-file function where the input data are retrieved from a MAT-file (Matlab data storage file format), the Fortran or C code called and executed and the output returned to the original MAT-file.

The second approach is more efficient but requires more work since it also require the creation of a gateway file to manage input and output data, and the compilation of the final routine into a Matlab MEX-file (which is essentially a Fortran or C m-file).

It is not a goal of this document to cover in details the process to create shell escape functions or MEX-files. The detailed procedures necessary to integrate external codes are covered in the *Matlab External Interface Guide*, a manual that comes with the professional edition of Matlab.

On-Design Engine Optimization at Given Flight Conditions

This section covers step-by-step procedures to perform on-design optimization.

1. Go to the directory where the optimization and engine codes are.
2. Open a text editor window and start Matlab in another window.
3. Open, in the text editor, the file 'gaondes.m'.
4. While in the above file, set the scaled boundaries for the variable 'y' in the line calling the genetic algorithm. The example below demonstrates which line to go to and where are the positions of each design variable bounds. The values for 'y' are between zero and 10, except for y_5 , the bypass ratio, which can be higher. The conversion between the scaled design variables, 'y', and their proper value, 'x', is as follows:

$$\begin{aligned}
y_1 * 10 &= x_1 = \Pi_c \\
y_2 &= x_2 = \Pi_c' \\
y_3 * 1000 &= x_3 = T_{t4} \\
y_4 * 1000 &= x_4 = T_{t7} \\
y_5 / 10 &= x_5 = \alpha \\
y_6 * 100 &= x_6 = m_0 \\
y_7 / 100 &= x_7 = C_{TO} \\
y_8 / 10 &= x_8 = M_5
\end{aligned}$$

Example:

$$\begin{array}{cccccccc}
y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & y_7 & y_8 \\
\downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\
\hline
y = \text{ga}([2 \ 3; 1.5 \ 5; 1.5 \ 3.2; 3.2 \ 3.6; 2 \ 10; 1.9 \ 2.2; 1 \ 3; 2 \ 9], 'gafunbis')
\end{array}$$

5. Save file ('gaondes.m') and close.
 6. Open file 'gaonx.m' in the text editor.
 7. At the beginning of file 'gaonx.m', set, as desired, the parameters C_{pc} , C_{pt} , γ_c , γ_t , h_{pr} , ε_1 , ε_2 , Π_b , Π_{dmax} , Π_N , e_c' , e_{cH} , e_{tH} , e_{tL} , η_{mPTO} , η_{mL} , η_{mH} , β , Π_{AB} , η_{AB} , γ_{AB} , Π_{mixmax} , C_{pAB} , P_0/P_9 , M_0 , h , η_b , T_{t3} , P_{t3} , Low and High pressure spools rpms, P_{TO} , K_1 , K_2 , K_3 , C_1 , C_2 , C_3 , m_{0spec} , W_{eng}/m_0 , and r_{hub}/r_{tip} . Refer to Annex A for the computer code names corresponding to these parameters.
 8. At the top of the second page of file 'gaonx.m', set the boundaries of the linear scaling functions. These boundaries represent the absolute lower and upper limits expected for the specific fuel consumption (1/hr), the aircraft cost (\$) and the engine annulus area (ft²). See Annex A for the proper variable names. The bnd1### boundary applies to the most desirable limit while the bnd0### applies to the least desirable.
- Example. If the aircraft cost is expected to fall between \$10,000,000 and \$30,000,000, then bnd1Cost = 10,000,000 (since a low cost is desirable) and bnd0Cost = 30,000,000.
9. Save and close file.
 10. Open file 'gafunbis.m' in the text editor.
 11. At the beginning of the file, set value for W_A , W_P , W_{TO} , and W_f . See Annex A for the proper variable names.

12. At the bottom of the first page of 'gafunbis.m', set the used-defined constraints limits. All constraints equations are of the form 'ctg# = expression'. The user-defined constraints equations are as follows, and the values to be set by the user are indicated with bold font:

```
ctg4 = Pt5primePt5 - Pt5/Pt5 max;
ctg6 = M5prime - M5 max;
ctg7 = - M5prime + M5 min;
ctg8 = ((P0*pir*pid*picprime*pich)/144) - Pt3 max (psi);
ctg9 = (T0*taur*taucprime*tauch) - Tt3 max (R);
ctg10 = ((Cpt*Tt4*etath*(1-(pith^((gammat-1)/gammat))))/(Tt4/518.7)) - ΔHT/θ max;
                                                    (BTU/lbm)

ctg11 = - F + F min (lb);
ctg12 = Cost - Cost max ($);
ctg13 = ((Area/((1-(rhubrtip^2))*pi))^0.5) - r max (ft);
ctg14 = Wf - Wf max;
ctg15 = - Wf + Wf min;
```

Note: to remove a constraint, just add '%' at the beginning of the constraint equation code line and in front of all code lines of the appropriate penalty 'IF' loop (i.e. in front of 'if ...', 'penal# = ...', and 'end').

13. Adjust the penalty functions (in 'gafunbis.m') with the constraints values set in step 12 above. The penalties equations are one the second and third pages. The user-defined penalties equations are as follows, and the values to be set by the user are indicated with bold font:

```
penal4 = (1.5*((ctg4 + Pt5/Pt5 max) / Pt5/Pt5 max))^2;
penal6 = (1.5*((ctg6 + M5 max) / M5 max))^2;
penal7 = (1.5*((ctg7 + M5 min) / M5 min))^2;
penal8 = (1.5*((ctg8 + Pt3 max (psi)) / Pt3 max (psi)))^2;
penal9 = (1.5*((ctg9 + Tt3 max (R)) / Tt3 max (R)))^2;
penal10 = (1.5*((ctg10 + ΔHT/θ max (BTU/lbm)) / ΔHT/θ max (BTU/lbm)))^2;
penal11 = ((ctg11 + F min (lb)) / F min (lb))^3;
penal12 = (1.5*((ctg12 + Cost max ($)) / Cost max ($)))^2;
penal13 = (1.5*((ctg13 + r max (ft)) / r max (ft)))^2;
penal14 = (1.5*((ctg14 + Wf max) / Wf max))^2;
penal15 = (1.5*((ctg15 + Wf min) / Wf min))^2;
```

14. Save and close file.

15. Open file 'fun.m' in text editor. Repeat steps 11 and 12. In this file, the constraints functions are expressed as array value g(x) (instead of 'ctg# = ...'). If some constraints are disabled, it is very **important** to renumber the active constraints **sequentially** from g(1) to g(k). Save and close the file.

16. Open file 'onxfinal.m' in text editor. Set engine parameters as per step 7, then save and close file.
17. Start optimization process by typing 'gaondes' in Matlab.
18. Once the process is completed, cut and paste all the desired outputs to another file.

Engine Optimization with Mission

This section covers step-by-step procedures to perform engine optimization with mission.

1. Go to the directory where the optimization and engine codes are.
2. Open a text editor window and start Matlab in another window.
3. With the text editor, create a drag profile '.m' file which evaluate K_1 , K_2 , C_{D0} and C_D for different Mach numbers. For the sake of standardization the file name should be of the form 'drag###.m'. See file 'dragsri.m' and 'draggsa.m' for examples of such files. Save and close the file.
4. Create a mission profile m-file with the text editor. . For the sake of standardization the file name should be of the form 'leg###.m'.

First, set the number of mission legs with the variable 'nleg'. Second, set the mission profile array which contains the type of legs and the off-design and leg parameters for each leg. The array has dimensions nleg x 14 and uses the following format:

$$L = \begin{bmatrix} a_1 & b_1 & c_1 & d_1 & e_1 & f_1 & g_1 & h_1 & i_1 & j_1 & k_1 & l_1 & m_1 & n_1; \\ a_2 & b_2 & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & m_2 & n_2; \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots; \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots; \\ a_{nleg} & b_{nleg} & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & n_{nleg} \end{bmatrix}$$

where the parameters for L are as follows:

a_i = leg type code
 b_i = initial Mach number (initial velocity (ft/s) for constant energy height maneuver)
 c_i = final Mach number (final velocity (ft/s) for constant energy height maneuver)
 d_i = initial altitude (ft)
 e_i = final altitude (ft)
 f_i = number of g's for turn legs, if applicable
 g_i = number of 360° turns for turn legs, if applicable
 h_i = distance (nm) or time (sec); time in minutes for loiter
 i_i = afterburner setting; if zero, AB=0 (off); if one AB=1 (on)
 j_i = expendable weight delivered (lb), if applicable
 k_i = angle of descent (deg) if applicable
 $l_i = T_{t4of} (R)$
 $m_i = T_{t7of} (R)$
 $n_i = P_0/P_{9of}$

The leg type codes are as follows:

1 = constant speed climb
 2 = horizontal acceleration
 3 = climb and acceleration
 4 = takeoff acceleration
 5 = constant altitude/speed cruise
 6 = constant altitude/speed turn
 7 = best cruise Mach number and altitude cruise
 8 = loiter
 9 = warmup
 10 = takeoff rotation
 11 = constant energy height maneuver
 12 = deliver expendables
 13 = descent

Third, Set the variable 'Maxreq' to one if there is a requirement for a maximum Mach/maximum altitude leg. Set 'Maxreq' to zero if there is no such requirement. If 'Maxreq' is set to one, off-design data for the maximum Mach/maximum altitude leg must be provided as follows:

$hofmach$ = maximum altitude (ft);
 $M0ofmach$ = maximum Mach number;
 $ABmach$ = afterburner setting for the leg (= 1 if on, = 0 if off)
 $Tt4ofmach = T_{t4}$ for the leg (R);
 $Tt7ofmach = T_{t7}$ for the leg (R);
 $P0P9ofmach = P_0/P_9$ for the leg;

Example. A 3-leg mission includes a cruise leg at $M = 1.2$ and $h = 30,000$ ft with afterburner off, a 4g turn at $M = 0.9$ and $h = 30,000$ ft with afterburner on, and a cruise

leg at $M = 1.4$ and $h = 30,000$ ft with afterburner off. There is a requirement for a maximum Mach/maximum altitude leg at $M = 2.5$ and $h = 50,000$ ft with afterburner on. The mission profile m-file would look like this:

```
Maxreq=1; % Switch to indicate there is a maximum Mach leg

% Maximum Mach leg off design data

hofmach=50000;
M0ofmach=2.5;
ABmach=1;
Tt4ofmach=3200;
Tt7ofmach=3600;
POP9ofmach=1;

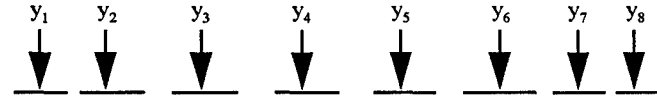
nleg=3;

L=[5 1.2 1.2 30000 30000 0 0 275 0 0 0 3200 3600 1;
  6 0.9 0.9 30000 30000 4 2 0 1 0 0 3200 3600 1;
  5 1.4 1.4 30000 30000 0 0 200 0 0 0 3200 3600 1];
```

5. Save and close mission profile.
6. Open, in the text editor, the file 'gaoptmiss.m'.
7. While in the above file, set the scaled boundaries for the variable 'y' in the line calling the genetic algorithm. The example below demonstrates which line to go to and where are the position of each design variable bounds. The values for 'y' are between zero and 10, except for y_5 , the bypass ratio, which can be higher. The conversion between the scaled design variables, 'y', and their proper value, 'x', is as follows:

$$\begin{aligned}
 y_1 * 10 &= x_1 = \Pi_c \\
 y_2 &= x_2 = \Pi_c \\
 y_3 * 1000 &= x_3 = T_{t4} \\
 y_4 * 1000 &= x_4 = T_{t7} \\
 y_5 / 10 &= x_5 = \alpha \\
 y_6 * 100 &= x_6 = m_0 \\
 y_7 / 100 &= x_7 = C_{TO} \\
 y_8 / 10 &= x_8 = M_5
 \end{aligned}$$

Example:

y_1 y_2 y_3 y_4 y_5 y_6 y_7 y_8


$[y, \dots] = \text{ganadon}([2 \ 3; 1.5 \ 5; 1.5 \ 3.2; 3.2 \ 3.6; 2 \ 10; 1.9 \ 2.2; 1 \ 3; 2 \ 9], 'gafunmiss', \dots)$

8. Save file ('gaoptmiss.m') and close.
9. Open file 'onxopt.m' in the text editor.
10. At the beginning of file 'gaonx.m', set, as desired, the parameters C_{pc} , C_{pt} , γ_c , γ_t , h_{pr} , ϵ_1 , ϵ_2 , Π_b , Π_{dmax} , Π_N , e_c , e_{cH} , e_{tH} , e_{tL} , η_{mPTO} , η_{mL} , η_{mH} , β , Π_{AB} , η_{AB} , γ_{AB} , Π_{mixmax} , C_{pAB} , P_0/P_9 , M_0 , h , η_b , T_{t3} , P_{t3} , Low and High pressure spools rpms, P_{TO} , K_1 , K_2 , K_3 , C_1 , C_2 , C_3 , m_{0spec} , W_{eng}/m_0 , and r_{hub}/r_{tip} . Refer to Annex A for the computer code names corresponding to these parameters.
11. At the bottom of the second page of file 'onxopt.m', set the boundaries of the linear scaling functions. These boundaries represent the absolute lower and upper limits expected for the fuel weight (lb), the aircraft cost (\$) and the engine annulus area (ft²). See Annex A for the proper variable names. The bnd1### boundary applies to the most desirable limit while the bnd0### applies to the least desirable.

Note: optimization with mission is set with sea-level static as the design point.

Example. if the aircraft cost is expected to fall between \$10,000,000 and \$30,000,000, then bnd1Cost = 10,000,000 (since a low cost is desirable) and bnd0Cost = 30,000,000.

12. Save and close file.
13. Open file 'gamiss.m' in the text editor.
14. At the beginning of the file 'gamiss.m', set, as desired, the mission and parameters W_{TO}/S , T_{sl}/W_{TO} , C_{DRTO} , a_{sl} , C_{Lmax} , μ_{TO} , r_{mc} , a , b , M_{crit} , k_{TO} , % installation losses, number of engines, M , maximum takeoff distance, W_P , W_{PE} , and W_A . Refer to Annex A for the computer code names corresponding to these parameters.

Note: If the drag profile involves a value of K_2 different than zero, insert '%' at the beginning of the code line ' $K_2=0$ '. This code line is at the top of the parameters code lines.

15. At the end of the second page, replace the 'leg###' mission line with the proper mission profile file name. The location of this code line in 'gamiss.m' is as follows:

```
%
% Calling mission profile using leg###.m
% It contains mission legs data and number of mission leg
% There is one file per aircraft/mission
%
legsri  ← mission profile code line to change to proper mission profile file name
%
% beginning of mission analysis loop
```

16. Save and close 'gamiss.m'.

17. Open all mission leg files applicable to the current mission profile. The mission legs and their corresponding files are as follows:

Constant speed climb: 'csspeedclb.m'

Horizontal acceleration: 'horizaccel.m'

Climb and acceleration: 'clbaccel.m'

Takeoff acceleration: 'tkoffaccel.m'

Constant altitude/speed cruise: 'csalspcrs.m'

Constant altitude/speed turn: 'csalsptrn.m'

Best cruise Mach number and altitude cruise: 'bcmmbca.m'

Loiter: 'loiter.m'

Warm-up: 'warmup.m'

Takeoff rotation: 'takofrot.m'

Constant energy height maneuver: 'csenhgt.m'

Deliver expendables: 'delivexp.m'

Descent: 'descent.m'

18. On the first page of each applicable leg file, replace the 'drag####' drag profile line with the proper drag profile file name. The location of this code line in all leg files is as follows:

%

% Evaluating Cdo, K1 and CD with the given drag profile drag####.m

%

dragsri \Leftarrow drag profile code line to change to proper drag profile file name

%

19. Save and close all leg files.

20. Open file 'gafunmiss.m' in the text editor.

21. At the top of the third page of 'gafunmiss.m', set the used-defined constraints limits. All constraints equations are of the form 'ctg# = expression'. The user-defined constraints equations are as follows, and the values to be set by the user are indicated with bold font:

ctg4 = $P_{t5prime} P_{t5} - P_{t5}$ **max**;

ctg6 = $M_{5prime} - M_5$ **max**;

ctg7 = $-M_{5prime} + M_5$ **min**;

ctg8 = $((P_0 * p_{ir} * p_{id} * p_{icprime} * p_{ich}) / 144) - P_{t3}$ **max (psi)**;

ctg9 = $(T_0 * t_{aur} * t_{aucprime} * t_{auch}) - T_{t3}$ **max (R)**;

ctg10 = $((C_{pt} * T_{t4} * e_{tath} * (1 - (p_{ith}^{((gammat-1)/gammat)}))) / (T_{t4} / 518.7)) - \Delta H_T / \theta$ **max;**
(BTU/lbm)

```

ctg11 = - F + F min (lb);
ctg12 = Cost - Cost max ($);
ctg13 = ((Area/((1-(rhbrtip^2))*pi))^0.5) - r max (ft);
ctg14 = Wf - Wf max;
ctg15 = - Wf + Wf min;
ctg16 = - MinWeight + Wmin

```

Note: to remove a constraint, just add '%' at the beginning of the constraint equation code line and in front of all code lines of the appropriate penalty 'IF' loop (i.e. in front of 'if ...', 'penal# = ...', and 'end').

22. Adjust the penalty functions (in 'gafunmiss.m') with the constraints values set in step 21 above. The penalties equations are one the third and fourth pages. The user-defined penalties equations are as follows, and the values to be set by the user are indicated with bold font:

```

penal4 = (1.5*((ctg4 + P15/P15 max) / P15/P15 max))^2;
penal6 = (1.5*((ctg6 + M5, max) / M5, max))^2;
penal7 = (1.5*((ctg7+ M5, min) / M5, min))^2;
penal8 = (1.5*((ctg8 + P13 max (psi)) / P13 max (psi)))^2;
penal9 = (1.5*((ctg9 + T13 max (R)) / T13 max (R)))^2;
penal10 = (1.5*((ctg10 + ΔHT/θ max (BTU/lbm)) / ΔHT/θ max (BTU/lbm)))^2;
penal11 = ((ctg11 + F min (lb)) / F min (lb))^3;
penal12 = (1.5*((ctg12 + Cost max ($)) / Cost max ($)))^2;
penal13 = (1.5*((ctg13 + r max (ft)) / r max (ft)))^2;
penal14 = (1.5*((ctg14 + Wf max) / Wf max))^2;
penal15 = (1.5*((ctg15 + Wf min) / Wf min))^2;
penal16 = (1.5*((ctg16 + Wmin) / Wmin))^2;

```

23. Save and close file 'gafunmiss.m'.

24. Open file 'funmiss2.m' in text editor. Repeat step 21. In this file, the constraints functions are expressed as array value g(x) (instead of 'ctg#...'). If some constraints are disabled, it is very **important** to renumber the active constraints **sequentially** from g(1) to g(k). Save and close the file.

25. Open file 'missfinal.m' in text editor. Set engine parameters as per steps 14 and 15, then save and close file.

26. Open file 'gaonxfinal.m' in text editor. Set engine parameters as per step 10, then save and close file.

27. Open file 'ganadon.m' in text editor. On the first page, right below the comments, set the initial values of the gross weight, the variable 'Wto', and the initial value of the gross weight for the best design, the variable 'Wtobest' (it must be the same value as

'Wto'). This step is not necessary if you use a weight model which determine a value of W_{TO} for each design, as describe in Chapter 2, equations (14) and (15) of the related thesis. If such a model is used, a value of takeoff weight without fuel, variable W_{empty} , must be given at the beginning of 'gaoptmiss.m', 'gafunmiss.m' and 'funmiss2.m'. The constraint 'ctg17' and its related penalty must be activated, as described in Steps 21 and 22. Moreover the gross weight models described in Step 28 below must be deactivated with 'symbols'.

28. In the same file, on the second last page, set your weight model. It should be an equation for the gross weight, W_{TO} , as a function of the fuel weight, W_F . Here are two examples of such models, as presently included in 'ganadon.m':

```
%
% Wto for GSA
%
WtoGAM=(1.3138634*Wfmin)+39621;    ⇐ Weight model for first aircraft
%
%Wto for SRI
%
%GAMMA=M*acoef*(1/(Wto^bcoef));
%WtoGAM=(Wp+(rmc*Wfmin))/(1-(Wfmin/Wto)-GAMMA);    ⇐ second aircraft
%
errwto=abs(Wto-WtoGAM);
%findata(4)=errwto;
%
%if errwto>100,
%
% For SRI
%
%Wto=(Wto+WtoGAM)/2;
%
%For GSA
%
Wto=WtoGAM;
%
```

29. Start optimization process by typing 'gaoptmiss' in Matlab.

30. Once the process is completed, cut and paste all the desired outputs to another file.

Cases with fixed W_{TO} . In some cases, it might be desirable to keep W_{TO} constant instead of updating it as W_F changes. Such a case could be the investigation of the payload or fuel weight difference between engine design. The procedure is the same as described above, with the following modifications:

- When the linear scaling parameters are set in 'onxopt.m', the boundaries for cost have to be the upper and lower limits of **engine** cost and **not** aircraft cost. The scaling

limits of all engine (i.e. for a given engine, the scaling values for a twin-engine plane are twice that of a single-engine one).

- The objective function in the file 'gafunmiss.m' must be adjusted for **engine** cost, **not** aircraft cost. On the second page of this file, '%' symbols have to be inserted at the beginning of the code lines 'costa = ...' and 'costb =... '.

- In the same file, the line 'Cost = (C1*F)+C2+(C3*costb);' has to be changed to 'Cost = C1*F; '.

- The modified portion of 'gafunmiss.m' should look like this:

```
%
% Evaluate cost
%
%costa=(F*(Wengm0))/Fm0;           <= '%' inserted here
%costb=Wto-Wf-Wp-Wa-costa;      <= '%' inserted here
Cost=C1*F;           <= cost term modified here
%
% Evaluate annulus area
%
```

- Modify the file 'ganadon.m' to disable the WTO convergence model. To do so add % symbols before the required code lines, located in the second last page of the file, as follows:

```
%if feasible==1,
%
%GAMMA=M*acoef*(1/(Wto^bcoef));
%WtoGAM=(Wp+(rmc*Wfmin))/(1-(Wfmin/Wto)-GAMMA);
%errwto=abs(Wto-WtoGAM);
%fndata(4)=errwto;
%
%if errwto>100,
%
%Wto=(Wto+WtoGAM)/2;
%fndata(1)=Wto;
%lowest=-10;
%fndata(2)=-10;
%Wfmin=1000000;
%fndata(3)=1000000;
%errwto=110;
%fndata(4)=errwto;
%feasible=0;
```

```

%fndata(5)=feasible;
%best1=[0 0 0 0 0 0 0 0];
%best2=[0 0 0 0 0 0 0 0];
%best3=[0 0 0 0 0 0 0 0];
%feasible=0;
%fndata(5)=feasible;
%
% end
%
%end

```

Stand-Alone On-Design Engine Cycle Analysis

This section covers step-by-step procedures to perform on-design engine cycle analysis.

1. Go to the directory where the optimization and engine codes are.
2. Open a text editor window and start Matlab in another window.
3. Open file 'onx.m' in the text editor.
4. At the beginning of file 'onx.m', set, as desired, the parameters C_{pc} , C_{pt} , γ_c , γ_t , h_{pr} , ε_1 , ε_2 , Π_b , Π_{dmax} , Π_N , e_c , e_{cH} , e_{tH} , e_{tL} , η_{mPTO} , η_{mL} , η_{mH} , β , Π_{AB} , η_{AB} , γ_{AB} , Π_{mixmax} , C_{pAB} , P_0/P_9 , M_0 , h , η_b , T_{t3} , P_{t3} , Low and High pressure spools rpms, P_{TO} , K_1 , K_2 , K_3 , C_1 , C_2 , C_3 , m_{0spec} , W_{eng}/m_0 , and r_{hub}/r_{tip} . Refer to Annex A for the computer code names corresponding to these parameters.
5. In Matlab, enter the desired design point as a 1 x 8 array. It should look like this:

$$x = [x_1 \ x_2 \ x_3 \ x_4 \ x_5 \ x_6 \ x_7 \ x_8]$$

where

```

x1 = value for  $\Pi_c$ 
x2 = value for  $\Pi_c$ 
x3 = value for  $T_{t4}$ 
x4 = value for  $T_{t7}$ 
x5 = value for  $\alpha$ 
x6 = value for  $m_0$ 
x7 = value for  $C_{TO}$ 
x8 = value for  $M_5$ 

```

6. Start on-design analysis process by typing 'onx' in Matlab.
7. Once the process is completed, cut and paste all the desired outputs to another file.

Stand-Alone Off-Design Engine Cycle Analysis

This section covers step-by-step procedures to perform on-design engine cycle analysis.

1. Go to the directory where the optimization and engine codes are.
2. Open a text editor window and start Matlab in another window.
3. Open file 'onx.m' in the text editor.
4. At the beginning of file 'onx.m', set, as desired, the parameters C_{pc} , C_{pt} , γ_c , γ_t , h_{pr} , ϵ_1 , ϵ_2 , Π_b , Π_{dmax} , Π_N , e_c , e_{cH} , e_{tH} , e_{tL} , η_{mPTO} , η_{mL} , η_{mH} , β , Π_{AB} , η_{AB} , γ_{AB} , Π_{mixmax} , C_{pAB} , P_0/P_9 , M_0 , h , η_b , T_{t3} , P_{t3} , Low and High pressure spools rpms, P_{TO} , K_1 , K_2 , K_3 , C_1 , C_2 , C_3 , m_{0spec} , W_{eng}/m_0 , and r_{hub}/r_{tip} . Refer to Annex A for the computer code names corresponding to these parameters.
5. In Matlab, enter the desired design point as a 1 x 8 array. It should look like this:

$$x = [x_1 \ x_2 \ x_3 \ x_4 \ x_5 \ x_6 \ x_7 \ x_8]$$

where

x_1 = value for Π_c
 x_2 = value for $\Pi_{c'}$
 x_3 = value for T_{t4}
 x_4 = value for T_{t7}
 x_5 = value for α
 x_6 = value for m_0
 x_7 = value for C_{TO}
 x_8 = value for M_5

6. In Matlab, enter the desired off-design conditions as a 1 x 6 array. It should look like this:

$$y = [y_1 \ y_2 \ y_3 \ y_4 \ y_5 \ y_6]$$

where

y_1 = value for M_0 , off-design
 y_2 = value for h , off-design
 y_3 = AB setting off-design (= 1, AB on; = 0, AB off)
 y_4 = value for P_0/P_9 off design
 y_5 = value for T_{t4} off-design
 y_6 = value for T_{t7} off-design

7. Start on-design analysis process by typing 'offx' in Matlab.
8. Once the process is completed, cut and paste all the desired outputs to another file.

Stand-Alone Mission Analysis

This section covers step-by-step procedures to perform mission analysis.

1. Go to the directory where the optimization and engine codes are.
2. Open a text editor window and start Matlab in another window.
3. With the text editor, create a drag profile m-file which evaluate K_1 , K_2 , C_{D0} and C_D for different Mach numbers. For the sake of standardization the file name should be of the form 'drag###.m'. See file 'dragsri.m' and 'draggsa.m' for examples of such files. Save and close the file.
4. Create a mission profile '.m' file with the text editor. . For the sake of standardization the file name should be of the form 'leg###.m'.

First, set the number of mission legs with the variable 'nleg'. Second, set the mission profile array which contains the type of legs and the off-design and leg parameters for each leg. The array has dimensions nleg x 14 and uses the following format:

$$L = \begin{bmatrix} a_1 & b_1 & c_1 & d_1 & e_1 & f_1 & g_1 & h_1 & i_1 & j_1 & k_1 & l_1 & m_1 & n_1; \\ a_2 & b_2 & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & n_2; \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots; \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots; \\ a_{nleg} & b_{nleg} & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & n_{nleg} \end{bmatrix}$$

where the parameters for L are as follows:

- a_i = leg type code
- b_i = initial Mach number (initial velocity (ft/s) for constant energy height maneuver)
- c_i = final Mach number (final velocity (ft/s) for constant energy height maneuver)
- d_i = initial altitude (ft)
- e_i = final altitude (ft)
- f_i = number of g's for turn legs, if applicable
- g_i = number of 360° turns for turn legs, if applicable
- h_i = distance (nm) or time (sec); time in minutes for loiter
- i_i = afterburner setting; if zero, AB=0 (off); if one AB=1 (on)
- j_i = expendable weight delivered (lb), if applicable

k_i = angle of descent (deg) if applicable

$l_i = T_{t4of}(R)$

$m_i = T_{t7of}(R)$

$n_i = P_0/P_{9of}$

The leg type codes are as follows:

- 1 = constant speed climb
- 2 = horizontal acceleration
- 3 = climb and acceleration
- 4 = takeoff acceleration
- 5 = constant altitude/speed cruise
- 6 = constant altitude/speed turn
- 7 = best cruise Mach number and altitude cruise
- 8 = loiter
- 9 = warm-up
- 10 = takeoff rotation
- 11 = constant energy height maneuver
- 12 = deliver expendables
- 13 = descent

Third, Set the variable 'Maxreq' to one if there is a requirement for a maximum Mach/maximum altitude leg. Set 'Maxreq' to zero if there is no such requirement. If 'Maxreq' is set to one, off-design data for the maximum Mach/maximum altitude leg must be provided as follows:

hofmach = maximum altitude (ft);

M0ofmach = maximum Mach number;

ABmach = afterburner setting for the leg (= 1 if on, = 0 if off)

Tt4ofmach = T_{t4} for the leg (R);

Tt7ofmach = T_{t7} for the leg (R);

P0P9ofmach = P_0/P_9 for the leg;

Example. A 3-leg mission includes a cruise leg at $M = 1.2$ and $h = 30,000$ ft with afterburner off, a 4g turn at $M = 0.9$ and $h = 30,000$ ft with afterburner on, and a cruise leg at $M = 1.4$ and $h = 30,000$ ft with afterburner off. There is a requirement for a maximum Mach/maximum altitude leg at $M = 2.5$ and $h = 50,000$ ft with afterburner on. The mission profile m-file would look like this:

Maxreq=1; % Switch to indicate there is a maximum Mach leg

% Maximum Mach leg off design data

hofmach=50000;
M0ofmach=2.5;
ABmach=1;
Tt4ofmach=3200;
Tt7ofmach=3600;
P0P9ofmach=1;

nleg=3;

L=[5 1.2 1.2 30000 30000 0 0 275 0 0 0 3200 3600 1;
6 0.9 0.9 30000 30000 4 2 0 1 0 0 3200 3600 1;
5 1.4 1.4 30000 30000 0 0 200 0 0 0 3200 3600 1];

5. Save and close mission profile.
6. Open file 'onx.m' in the text editor.
7. At the beginning of file 'onx.m', set, as desired, the parameters C_{pc} , C_{pt} , γ_c , γ_t , h_{pr} , ϵ_1 , ϵ_2 , Π_b , Π_{dmax} , Π_N , e_c , e_{cH} , e_{tH} , e_{tL} , η_{mPTO} , η_{mL} , η_{mH} , β , Π_{AB} , η_{AB} , γ_{AB} , Π_{mixmax} , C_{pAB} , P_0/P_9 , M_0 , h , η_b , T_{t3} , P_{t3} , Low and High pressure spools rpms, P_{TO} , K_1 , K_2 , K_3 , C_1 , C_2 , C_3 , m_{0spec} , W_{eng}/m_0 , and r_{hub}/r_{tip} . Refer to Annex A for the computer code names corresponding to these parameters.

Note: stand-alone mission analysis is set with sea-level static as the design point.

8. Open file 'miss.m' in the text editor.
9. At the beginning of the file 'miss.m', set, as desired, the mission and parameters W_{TO} , W_{TO}/S , T_{SL}/W_{TO} , C_{DRTO} , a_{SL} , C_{Lmax} , μ_{TO} , r_{mc} , a , b , M_{crit} , k_{TO} , % installation losses, number of engines, M , maximum takeoff distance, W_P , W_{PE} , and W_A . Refer to Annex A for the computer code names corresponding to these parameters.

Note: If the drag profile involves a value of K_2 different than zero, insert '%' at the beginning of the code line ' $K_2=0$ '. This code line is at the top of the parameters code lines.

10. At the beginning of the second page, replace the 'leg####' mission line with the proper mission profile file name. The location of this code line in 'miss.m' is as follows:

```

%
% Calling mission profile using leg###.m
% It contains mission legs data and number of mission leg
% There is one file per aircraft/mission
%
legsri    ⇐ mission profile code line to change to proper mission profile file name
%
% beginning of mission analysis loop
%

```

11. Save and close 'miss.m'.

12. Open all mission leg files applicable to the current mission profile. The mission legs and their corresponding files are as follows:

Constant speed climb: 'csspeedclb.m'
 Horizontal acceleration: 'horizaccel.m'
 Climb and acceleration: 'clbaccel.m'
 Takeoff acceleration: 'tkoffaccel.m'
 Constant altitude/speed cruise: 'csalspcrs.m'
 Constant altitude/speed turn: 'csalsptrn.m'
 Best cruise Mach number and altitude cruise: 'bcmbsca.m'
 Loiter: 'loiter.m'
 Warm-up: 'warmup.m'
 Takeoff rotation: 'takofrot.m'
 Constant energy height maneuver: 'csehgth.m'
 Deliver expendables: 'delivexp.m'
 Descent: 'descent.m'

13. On the first page of each applicable leg file, replace the 'drag###' drag profile line with the proper drag profile file name. The location of this code line in all leg files is as follows:

```

%
% Evaluating Cdo, K1 and CD with the given drag profile drag###.m
%
dragsri    ⇐ drag profile code line to change to proper drag profile file name
%

```

14. Save and close all leg files.

15. In Matlab, enter the desired design point as a 1 x 8 array. It should look like this:

$$x = [x_1 \ x_2 \ x_3 \ x_4 \ x_5 \ x_6 \ x_7 \ x_8]$$

where

x_1 = value for Π_c
 x_2 = value for Π_c
 x_3 = value for T_{t4}
 x_4 = value for T_{t7}
 x_5 = value for α
 x_6 = value for m_0
 x_7 = value for C_{TO}
 x_8 = value for M_5

16. Start on-design analysis process by typing 'miss' in Matlab.
17. Once the process is completed, cut and paste all the desired outputs to another file.

Computer Codes Variables

ABmach = Afterburner setting for maximum Mach/maximum altitude leg
 Area = Annulus area (ft²)
 asl = a_{SL} = Speed of sound at sea level (ft/s)
 acoef = a = Coefficient in Γ function equation
 bcoef = b = Coefficient in Γ function equation
 bnd1S = Linear scaling desirable limit for specific fuel consumption
 bnd0S = Linear scaling undesirable limit for specific fuel consumption
 bnd1Wf = Linear scaling desirable limit for fuel weight
 bnd0Wf = Linear scaling undesirable limit for fuel weight
 bnd1Cost = Linear scaling desirable limit for cost
 bnd0Cost = Linear scaling undesirable limit for cost
 bnd1Area = Linear scaling desirable limit for annulus area
 bnd0Area = Linear scaling undesirable limit for annulus area
 C1 = C_1 = Engine cost coefficient (\$/lb thrust)
 C2 = C_2 = Avionics cost coefficient (\$)
 C3 = C_3 = Airframe cost coefficient (\$/lb empty weight)
 CD = C_D = Drag coefficient
 CL = C_L = Lift coefficient
 CLmax = C_{Lmax} = Maximum lift coefficient
 Cpc = C_{pc} = Specific heat at constant pressure for compressor flow (BTU/lbm-R)
 Cpt = C_{pt} = Specific heat at constant pressure for turbine flow (BTU/lbm-R)
 Cto = C_{TO} = Power takeoff shaft power coefficient
 ctg# = constraint equation
 drag### = Drag profile m-file name
 ecprime = e_c = Polytopic efficiency of fan
 ech = e_{cH} = Polytopic efficiency of high pressure compressor
 eth = e_{tH} = Polytopic efficiency of high pressure turbine
 etl = e_{tL} = Polytopic efficiency of low pressure turbine
 F = F = Uninstalled thrust (lb)
 f = f = Fuel-to-air ratio of burner
 fAB = f_{AB} = Fuel-to-air ratio of afterburner
 f0 = f_0 = Overall engine fuel-to-air ratio
 h = h = Altitude (ft)
 hofmach = Altitude for maximum Mach/maximum altitude leg
 hpr = h_{PR} = Heating value of fuel
 Kk1 = K_1 = Objective weight factor for specific fuel consumption or fuel weight
 Kk2 = K_2 = Objective weight factor for cost
 Kk3 = K_3 = Objective weight factor for annulus area
 K1 = K_1 = Coefficient in lift-drag polar equation
 K2 = K_2 = Coefficient in lift-drag polar equation
 kto = k_{TO} = Velocity ratio at takeoff

leg### = Mission profile m-file name
 M = M = Material modifier
 M0ofmach = Mach number for maximum Mach/maximum altitude leg
 M5 = M₅ = Core flow Mach number at exhaust mixer
 Mcrit = M_{crit} = Drag rise critical Mach number
 Maxreq = Switch variable for maximum Mach/maximum altitude leg
 m0 = m₀ = Mass flow rate (lbm/s)
 m0spec = m_{0spec} = specific flow (lbm/sec-ft²)
 rpm1 = Rotational speed (rpm) limit for low pressure spool (% of design rpm)
 rpm2 = Rotational speed (rpm) limit for high pressure spool (% of design rpm)
 Neng = Number of engines
 Nt = N = Number of 360° turns
 ng = n = Load factor (g)
 nleg = Number of mission legs
 P0P9 = P₀/P₉ = Pressure ratio at nozzle exit
 P0P9ofmach = P₀/P₉ for maximum Mach/maximum altitude leg
 power = P_{TO} = Power of takeoff shaft (Watts)
 Pt3max = P_{t3max} = Total pressure limit at exit of compressor (psi)
 r = Engine radius (ft)
 rhubrtip = r_{hub}/r_{tip} = hub-to-tip ratio
 rmc = rmc = remaining fuel coefficient (% of total fuel)
 S = S = Uninstalled thrust specific fuel consumption (1/hr)
 T = T = Installed thrust (lb)
 TSFC = TSFC = Installed thrust specific fuel consumption (1/hr)
 TslWto = T_{SL}/W_{TO} = Thrust-to-weight ratio
 Tt3max = T_{t3max} = Total temperature limit at exit of compressor (R)
 Tt4 = T_{t4} = Total temperature at high pressure turbine entry (R)
 Tt7 = T_{t7} = Afterburner total temperature (R)
 Tt4ofmach = T_{t4} for maximum Mach/maximum altitude leg (R)
 Tt7ofmach = T_{t7} for maximum Mach/maximum altitude leg (R)
 takofdist = Takeoff distance (ft)
 tkofdistmax = Takeoff distance limit (ft)
 Wa = W_A = Avionics weight (lb)
 Wengm0 = W_{eng}/m₀ = Specific engine weight
 Wempty = Takeoff weight without fuel
 Wf = W_f = Fuel weight (lb)
 Wp = W_P = Payload weight (lb)
 Wpe = W_{PE} = Expended payload weight (lb)
 Wto = W_{TO} = Takeoff weight (lb)
 Wtobest = W_{TO} for best design (lb)
 WtoS = W_{TO}/S = Wing loading (lb/ft²)
 x = design variable vector (x₁ to x₈)
 y = scaled design variable vector (y₁ to y₈), off-design condition vector (y₁ to y₆)

$\alpha = \alpha$ = bypass ratio
 $\alpha' = \alpha'$ = bypass ratio at exhaust mixer
 $\beta_{\text{final}} = \beta_{\text{final}}$ = Weight fraction at end of mission
 $\beta = \beta$ = Bleed air fraction
 $\Delta H_T / \theta = \Delta H_T / \theta$ = High pressure turbine specific work (BTU/lbm)
 $\epsilon_1 = \epsilon_1$ = Cooling air #1 mass flow rate
 $\epsilon_2 = \epsilon_2$ = Cooling air #2 mass flow rate
 $\phi = \phi$ = Loss coefficient
 $\gamma_c = \gamma_c$ = Ratio of specific heat for compressor flow
 $\gamma_t = \gamma_t$ = Ratio of specific heat for turbine flow
 $\eta_{\text{AB}} = \eta_{\text{AB}}$ = Efficiency of afterburner
 $\eta_b = \eta_b$ = Efficiency of burner
 $\eta_c = \eta_c$ = Efficiency of fan
 $\eta_{\text{cH}} = \eta_{\text{cH}}$ = Efficiency of high pressure compressor
 $\eta_{\text{mH}} = \eta_{\text{mH}}$ = Power transfer efficiency of high pressure spool
 $\eta_{\text{mL}} = \eta_{\text{mL}}$ = Power transfer efficiency of low pressure spool
 $\eta_{\text{mP}} = \eta_{\text{mP}}$ = Power transfer efficiency of power takeoff shaft
 $\eta_{\text{tH}} = \eta_{\text{tH}}$ = Efficiency of high pressure turbine
 $\eta_{\text{tL}} = \eta_{\text{tL}}$ = Efficiency of low pressure turbine
 $\mu_{\text{TO}} = \mu_{\text{TO}}$ = Friction coefficient on takeoff
 $\Pi_{\text{AB}} = \Pi_{\text{AB}}$ = Total pressure ratio of afterburner
 $\Pi_b = \Pi_b$ = Total pressure ratio of burner
 $\Pi_c = \Pi_c$ = Total pressure ratio of compressor
 $\Pi_{\text{cH}} = \Pi_{\text{cH}}$ = Total pressure ratio of high pressure compressor
 $\Pi_c = \Pi_c$ = Total pressure ratio of fan
 $\Pi_d = \Pi_d$ = Total pressure ratio of diffuser (inlet)
 $\Pi_{\text{dmax}} = \Pi_{\text{dmax}}$ = Total pressure loss in diffuser (inlet) due to friction
 $\Pi_M = \Pi_M$ = Total pressure ratio of mixer
 $\Pi_{\text{Mmax}} = \Pi_{\text{Mmax}}$ = Total pressure loss in mixer due to friction
 $\Pi_n = \Pi_n$ = Total pressure ratio of nozzle
 $\Pi_r = \Pi_r$ = Isentropic freestream recovery pressure ratio
 $\Pi_{\text{tH}} = \Pi_{\text{tH}}$ = Total pressure ratio of high pressure turbine
 $\Pi_{\text{tL}} = \Pi_{\text{tL}}$ = Total pressure ratio of low pressure turbine
 $\tau_{\text{cH}} = \tau_{\text{cH}}$ = Total temperature ratio of high pressure compressor
 $\tau_c = \tau_c$ = Total temperature ratio of fan
 $\tau_M = \tau_M$ = Total temperature ratio of mixer
 $\tau_{\text{m1}} = \tau_{\text{m1}}$ = Total temperature ratio of station 4 to 4a
 $\tau_{\text{m2}} = \tau_{\text{m2}}$ = Total temperature ratio of station 4c to 4b
 $\tau_r = \tau_r$ = Adiabatic freestream recovery temperature ratio
 $\tau_{\text{tH}} = \tau_{\text{tH}}$ = Total temperature ratio of high pressure turbine
 $\tau_{\text{tL}} = \tau_{\text{tL}}$ = Total temperature ratio of low pressure turbine
 $\tau_\lambda = \tau_\lambda$ = Enthalpy ratio of burner
 $\tau_{\lambda\text{AB}} = \tau_{\lambda\text{AB}}$ = Enthalpy ratio of afterburner

Computer Codes File ListingStand-Alone On-Design Engine Cycle Analysis

onx.m : Performs stand-alone on-design cycle analysis
altitude.m : Evaluates altitude properties up to 65,000 ft

Stand-Alone Off-Design Engine Cycle Analysis

offx.m : Performs stand-alone off-design cycle analysis
onx.m : Performs stand alone on-design cycle analysis
altitudeof.m : Evaluates altitude properties for off-design conditions

Stand-Alone Mission Analysis

miss.m : Stand-alone mission analysis control program
offxsl.m : Evaluates sea-level static performance
leg###.m : Mission profile file
drag###.m : Drag profile file
legselect.m : Selects mission profile legs one at a time for evaluation
csspeedclb.m : Evaluates weight fraction for constant speed climb
horizaccel.m : Evaluates weight fraction for horizontal acceleration
clbaccel.m : Evaluates weight fraction for climb and acceleration
tkoffaccel.m : Evaluates weight fraction for takeoff acceleration
csalspcrs.m : Evaluates weight fraction for constant speed/altitude cruise
csalsptrn.m : Evaluates weight fraction for constant speed/altitude turn
bcmca.m : Evaluates weight fraction for best cruise Mach/altitude cruise
loiter.m : Evaluates weight fraction for loiter
warmup.m : Evaluates weight fraction for ground warmup
takofrot.m : Evaluates weight fraction for takeoff rotation
csehgts.m : Evaluates weight fraction for constant energy height maneuver
delivexp.m : Evaluates weight fraction when ordnance is dropped
descent.m : Evaluates weight fraction for descent
bestalti.m : Finds altitude for a given static pressure ratio up to 65,000 ft
offxmiss.m : Performs off-design analysis as part of mission analysis
throttle.m : Throttles back available thrust to required thrust

On-Design Engine Optimization at Given Flight Conditions

gaondes.m : On-design optimization control file
gaonx.m : Performs on-design analysis as part of on-design optimization
altitude.m : Evaluates altitude properties up to 65,000 ft
gafunbis.m : Global objective function file for on-design optimization
fun.m : Local objective function file for on design optimization
gaconvert.m : Converts scaled values of 'y' to 'x' as part of global optimization
convert.m : Converts scaled values of 'y' to 'x' as part of local optimization
ga.m : Genetic algorithm global optimizer control file for on-design
constr.m : Local gradient-based optimizer for on-design
onxfinal.m : Final on-design analysis for on-design optimization

Engine Optimization with Mission

gaoptmiss.m : Optimization with mission control file
ganadon.m : Genetic algorithm global optimizer control file for mission
gaconstr2.m : Local gradient-based optimizer for mission
gainit.m : Genetic algorithm initial population generator for mission
gafunmiss.m : Global objective function file for mission optimization
funmiss2.m : Local objective function file for mission optimization
gaconvert.m : Converts scaled values of 'y' to 'x' as part of global optimization
convert.m : Converts scaled values of 'y' to 'x' as part of local optimization
gamiss.m : Performs mission analysis for global mission optimization
gamiss2.m : Performs mission analysis for local mission optimization
leg###.m : Mission profile file
drag###.m : Drag profile file
legselect.m : Selects mission profile legs one at a time for evaluation
csspeedclb.m : Evaluates weight fraction for constant speed climb
horizaccel.m : Evaluates weight fraction for horizontal acceleration
clbaccel.m : Evaluates weight fraction for climb and acceleration
tkoffaccel.m : Evaluates weight fraction for takeoff acceleration
csalspcrs.m : Evaluates weight fraction for constant speed/altitude cruise
csalsptrn.m : Evaluates weight fraction for constant speed/altitude turn
bcmbsca.m : Evaluates weight fraction for best cruise Mach/altitude cruise
loiter.m : Evaluates weight fraction for loiter
warmup.m : Evaluates weight fraction for ground warmup
takofrot.m : Evaluates weight fraction for takeoff rotation
csenhgt.m : Evaluates weight fraction for constant energy height maneuver
delivexp.m : Evaluates weight fraction when ordnance is dropped
descent.m : Evaluates weight fraction for descent
bestalti.m : Finds altitude for a given static pressure ratio up to 65,000 ft
offxmiss.m : Performs off-design analysis as part of mission analysis
throttle.m : Throttles back available thrust to required thrust

onxopt.m : Performs on-design analysis as part of mission optimization
missfinal.m : Performs final mission analysis for mission optimization
gaonxfinal.m : Performs final on-design analysis for mission optimization

Bibliography

1. Arora, Jasbir S. Introduction to Optimum Design. New-York: McGraw-Hill, 1989.
2. Bertin, John J. and Smith, Michael L. Aerodynamics for Engineers. Englewood Cliffs, New-Jersey: Prentice-Hall, 1989.
3. Cramer, Evin J., Dennis J.E., Frank, Paul D., Lewis, Robert M. and Shubin' Gregory R. "Problem Formulation for Multidisciplinary Optimization," SIAM Journal of Optimization: 754-776 (November 1994).
4. Fredette, Ray E. "Beyond IHPTET Aircraft Payoff Study," Report to Wright Laboratories, Aero Propulsion and Power Directorate, Turbine Engine Division, Wright-Patterson AFB OH. July 1996.
5. Goldberg, David E. Genetic Algorithm in Search, Optimization and Machine Learning. Reading, Massachusetts: Addison-Wesley Publishing Company, 1989.
6. Grace, Andrew. Optimization Toolbox User's Guide. Natick, Massachussets: The Mathworks, 1992.
7. Houck, Christopher R., Joines, Jeffery A. and Kay, Michael G. "A Genetic Algorithm for Function Optimization: A Matlab Implementation". North Carolina State University.
8. King, Paul I. Class handout, Meng 733, Airbreathing Engine Design. School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB OH, March 1996.
9. Lach, Allan. Development of an Engine/Airframe Matching Scheme for Jet Engine Retrofit. MS thesis, AFIT/GAE/ENY/92D-19. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, December 1992 (AAJ-4597).
10. Mattingly, Jack D., Heiser, William H. and Daley Daniel H. Aircraft Engine Design. New-York: American Institute of Aeronautics and Astronautics, 1987.
11. Mattingly, Jack D. MISS, Mission Analysis Program User Guide, Version 2.1, October 1989. Provided by Professor Paul King. Air Force Institute of Technology, Wright-Patterson AFB OH, March 1996.
12. -----, On-Design and Off-Design Aircraft Engine Cycle Analysis Programs. Washington, D.C.: American Institute of Aeronautics and Astronautics, 1990.

13. Netzer, Dave and Jenson, Gordon E. Tactical Missile Propulsion. Washington, D.C.: American Institute of Aeronautics and Astronautics, 1996.
14. Rao, Singiresu S. Engineering Optimization. New-York: John Wiley and Sons, 1996.
15. Stricker, Jeff M. "The Gas Turbine Conceptual Design Process - an Integrated Approach." Paper submitted for The 1996 Turbine Engine Technology Symposium, Dayton, OH. July 1996.
16. -----. Group Leader Wright Laboratories, Aero Propulsion and Power Directorate, Turbine Engine Division, Wright-Patterson AFB OH. Personal interview. 3 July 1996.

Vita

Captain Luc Nadon was born [REDACTED] [REDACTED], Quebec, Canada. He graduated from Ecole Secondaire Le Campus high school in 1983. He entered undergraduate studies at the Collège Militaire Royal de St-Jean, Quebec and the Royal Military College in Kingston, Ontario. He graduated with a Bachelor of Science in Mechanical Engineering in May 1988. He received his commission on May 14, 1988 and then completed his basic aerospace engineering training at CFB Borden, Ontario. His first assignment was at CFB Bagotville, Quebec, first as the Base Armament and Photo officer and then as an Aircraft Maintenance Officer in a CF-18 squadron. Capt Nadon next assignment was at CFB Borden as an armament and engineering instructor. In May, 1995, he entered the School of Engineering, Air Force Institute of Technology.

[REDACTED]
Permanent Address

[REDACTED]
St-Basile

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.				
1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE December 1996	3. REPORT TYPE AND DATES COVERED Master's Thesis		
4. TITLE AND SUBTITLE Multidisciplinary and Multiobjective Optimization in Conceptual Design for Mixed-Stream Turbofan engines			5. FUNDING NUMBERS	
6. AUTHOR(S) Luc J.J.P. Nadon, Capt, CAF				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Air Force Institute of Technology, WPAFB OH 45433-6583			8. PERFORMING ORGANIZATION REPORT NUMBER AFIT/GAE/ENY/96D-6	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Wright Laboratory WL/POTA Bldg 18, Area B WPAFB OH 45433-7251			10. SPONSORING/MONITORING AGENCY REPORT NUMBER	
11. SUPPLEMENTARY NOTES				
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution unlimited			12b. DISTRIBUTION CODE A	
13. ABSTRACT (Maximum 200 words) Despite major advances in design tools such as engine cycle analysis software and computer-aided design, conceptual gas turbine engine design is essentially a trial-and-error process based on the experience of engineers. Modern optimization concepts, such as multidisciplinary optimization, and multiobjective optimization, linked with sequential quadratic programming (SQP) methods and genetic algorithms (GA), were applied to the conceptual engine design process to automate the conceptual design phase. Robust integrated computer codes were created to find the optimal values of eight engine parameters in order to minimize fuel usage, aircraft cost and engine annulus area over a given mission. The engine cycle selected for study was the mixed-stream, low-bypass turbofan. SQP and GA optimization algorithms were integrated with on-design and off-design engine cycle analysis and mission analysis computer codes created by the authors to obtain the optimized conceptual engine design for an imaginary short range interceptor and the Global Strike Aircraft U.S. Air Force concept. The process used a non-specific approach that can be applied to a wide variety of missions and aircraft. All the codes were written in Matlab, and so operate under the same programming architecture and can be easily upgraded or modified.				
14. SUBJECT TERMS Optimization, Propulsion Systems, Turbomachinery, Aircraft Engine, Low-Bypass Turbofan			15. NUMBER OF PAGES 170	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL	

GENERAL INSTRUCTIONS FOR COMPLETING SF 298

The Report Documentation Page (RDP) is used in announcing and cataloging reports. It is important that this information be consistent with the rest of the report, particularly the cover and title page. Instructions for filling in each block of the form follow. It is important to *stay within the lines* to meet optical scanning requirements.

Block 1. Agency Use Only (Leave blank).

Block 2. Report Date. Full publication date including day, month, and year, if available (e.g. 1 Jan 88). Must cite at least the year.

Block 3. Type of Report and Dates Covered. State whether report is interim, final, etc. If applicable, enter inclusive report dates (e.g. 10 Jun 87 - 30 Jun 88).

Block 4. Title and Subtitle. A title is taken from the part of the report that provides the most meaningful and complete information. When a report is prepared in more than one volume, repeat the primary title, add volume number, and include subtitle for the specific volume. On classified documents enter the title classification in parentheses.

Block 5. Funding Numbers. To include contract and grant numbers; may include program element number(s), project number(s), task number(s), and work unit number(s). Use the following labels:

C - Contract	PR - Project
G - Grant	TA - Task
PE - Program Element	WU - Work Unit Accession No.

Block 6. Author(s). Name(s) of person(s) responsible for writing the report, performing the research, or credited with the content of the report. If editor or compiler, this should follow the name(s).

Block 7. Performing Organization Name(s) and Address(es). Self-explanatory.

Block 8. Performing Organization Report Number. Enter the unique alphanumeric report number(s) assigned by the organization performing the report.

Block 9. Sponsoring/Monitoring Agency Name(s) and Address(es). Self-explanatory.

Block 10. Sponsoring/Monitoring Agency Report Number. (If known)

Block 11. Supplementary Notes. Enter information not included elsewhere such as: Prepared in cooperation with...; Trans. of...; To be published in.... When a report is revised, include a statement whether the new report supersedes or supplements the older report.

Block 12a. Distribution/Availability Statement. Denotes public availability or limitations. Cite any availability to the public. Enter additional limitations or special markings in all capitals (e.g. NOFORN, REL, ITAR).

DOD - See DoDD 5230.24, "Distribution Statements on Technical Documents."

DOE - See authorities.

NASA - See Handbook NHB 2200.2.

NTIS - Leave blank.

Block 12b. Distribution Code.

DOD - Leave blank.

DOE - Enter DOE distribution categories from the Standard Distribution for Unclassified Scientific and Technical Reports.

NASA - Leave blank.

NTIS - Leave blank.

Block 13. Abstract. Include a brief (*Maximum 200 words*) factual summary of the most significant information contained in the report.

Block 14. Subject Terms. Keywords or phrases identifying major subjects in the report.

Block 15. Number of Pages. Enter the total number of pages.

Block 16. Price Code. Enter appropriate price code (*NTIS only*).

Blocks 17. - 19. Security Classifications. Self-explanatory. Enter U.S. Security Classification in accordance with U.S. Security Regulations (i.e., UNCLASSIFIED). If form contains classified information, stamp classification on the top and bottom of the page.

Block 20. Limitation of Abstract. This block must be completed to assign a limitation to the abstract. Enter either UL (unlimited) or SAR (same as report). An entry in this block is necessary if the abstract is to be limited. If blank, the abstract is assumed to be unlimited.